Ghent University
Faculty of Sciences
Department of Molecular Genetics

VIB, Flanders Institute for Biotechnology
Department of Plant Systems Biology
Computational Biology division

Biological knowledge management and gene network analysis: a heuristic road to Systems Biology

Steven Vercruysse

Promotors:
Prof. Dr. Martin Kuiper and Prof. Dr. Yves Van de Peer

2008
Dissertation submitted in fulfillment of the requirements for the degree of
Doctor in Sciences, Biotechnology

It is not the mountain
we conquer but ourselves.

Table of Contents

Summary in English

In order to understand the molecular basis of living cells and organisms, biologists over the past decades have been studying life's core molecular players: the *genes*. Most genes have a specific *function*, a role they play in the collective task of developing a cell and supporting all the aspects of keeping it alive. These genes do not perform their function randomly. Instead, after billions of years of evolution, nature's trial-and-error process, they have become parts of an utterly complex and intricate network, an interconnected mesh of genes that comprises signal detection cascades, enzymatic reactions, control mechanisms, etc. Over several past decades, experimental molecular biologists have sought mainly to study these genes via a one-by-one approach. However, with the advent of high-throughput experimental techniques, the number-crunching power of computers, and the realisation that many biological functions are the result of interactions between genes or their proteins, Biology's related field of *Systems Biology* has emerged. Here, one tries to combine the dispersed information produced by many researchers, in integrated assemblies called *gene networks*.

Our research comprises the development of two new methods for improved information integration in the field of molecular Systems Biology. The first one aims to support an approach to acquire insights in the dynamics of gene networks (the behaviour of gene activities over time), called 'modelling and simulation' of genetic regulatory networks. Our second new method approaches the problem of how to collect and manage the information necessary to compose such genetic networks in the first place, based on scattered information in a dispersed and increasingly fast growing body of publications. These two methods form two separate parts in this thesis (chapters 2-4, and chapters 5-7).

Chapter 1, section 1.3 provides an introductory, complete overview of this thesis. It is intended as a light introduction to my doctoral research, presented in an informal and entertaining way, and mainly addressed to my friends and family. It forms an introduction for the laymen to our work and the concepts that are important for this thesis.

Chapters 2, 3 and 4 constitute Part 1 of this thesis. Chapter 2 gives a review of the various formalisms for modelling and simulation of gene networks, as a thorough background for our work presented in the following chapter. Chapter 3 describes SIM-plex, our new software tool that forms a bridge between a mathematical gene network modelling formalism, and the biologist, who usually is more an expert in the biology behind the gene network than a mathematician can ever be. It shields off the mathematics in a new way so as to enable biologists to experiment with modelling and simulation themselves. Chapter 4 describes the various applications that SIM-plex was used for.

The research described in Part 2 of this thesis, chapters 5, 6 and 7, emerged from our own need for a better management of biological information. We experienced this necessity while we were building a larger genetic network for the Arabidopsis cell cycle, and it forms a general problem in biology. Chapter 5 gives a background of the currently existing methods for harvesting literature information, but comes to the conclusion that no existing

automated or manual method displays sufficient potential to capture the largest part of information from literature in a structured way. In chapter 6, we describe our bold proposal of a new method to tackle this problem: MineMap, a community-based manual text-curation initiative. We describe the various aspects required to make such a project possible, based on our own experiences with our prototype application MineMap. This research is organised in a 'heuristic' way, in the sense that we built a first sketch and a working solution that also generated experiences for improvements in a next design. While chapter 6 describes our new ideas and concrete implementations in considerable detail, chapter 7 then illustrates the core concept behind MineMap.

Samenvatting in het Nederlands

Titel in het Nederlands: "Biologische kennis beheer, en gen-netwerk analyse: een heuristische route naar Systeembiologie."

Om te begrijpen hoe cellen en organismen werken op het moleculaire niveau, hebben biologen de voorbije decennia de moleculaire hoofdrolspelers van het leven bestudeerd: de *genen*. De meeste genen hebben een specifieke *functie*, een rol die ze spelen in hun gezamenlijke taak om een cel te ontwikkelen, en voor het ondersteunen van alle aspecten om ze in leven te houden. Deze genen voeren hun functie helemaal niet willekeurig of continu uit. In tegendeel, want na miljarden jaren van evolutie, het probeer-en-corrigeer proces van de natuur, zijn de onstane genen deel geworden van een uiterst complex en ingewikkeld netwerk. Het is een uitgebreid schakelwerk van genen dat samen functies verzorgt zoals signaaldetectie-cascades, enzymatische reacties, controlemechanismen, enz. Gedurende decennia hebben experimenteel-moleculaire biologen deze genen vooral één per één bestudeerd. Maar ondertussen, dankzij de komst van hoge-doorvoer massa-experimenten, de rekenkracht van computers, en de realisatie dat vele biologische functies het resultaat zijn van interacties tussen verscheidene genen en hun proteïnen, is een gerelateerd veld van de biologie ontstaan: Systeembiologie. Hier tracht men de verspreide informatie, geproduceerd door vele onderzoekers, te monteren in geïntegreerde overzichten genaamd *gen-netwerken*.

Ons onderzoek omvat de ontwikkeling van twee nieuwe methodes voor een verbeterde informatie-integratie in de Systeembiologie. De eerste ondersteunt een aanpak om inzicht te krijgen in de dynamiek van gen-netwerken (het gedrag van gen-activiteiten in functie van de tijd), genaamd het 'modelleren en simuleren' van gen-netwerken. Onze tweede nieuwe methode benadert het probleem om de overvloed aan gegevens, nodig om zulke gen-netwerken samen te stellen, in de eerste plaats ook te kunnen verzamelen en beheren. Dit is geen evidente taak want deze informatie is her en der verspreid in een steeds groter wordende verzameling van miljoenen wetenschappelijke publicaties. Deze twee methodes vormen twee aparte delen in deze thesis: hoofdstukken 2-4, en hoofdstukken 5-7.

Hoofdstuk 1, sectie 1.3 begint met een inleidend, volledig overzicht op deze thesis. Het is bedoeld als een lichte introductie tot mijn doctoraal onderzoek, gepresenteerd op een informele en aangename manier, en vooral gericht tot vrienden en familie (die Engels begrijpen). Het vormt een inleiding voor buitenstaanders tot ons werk en de belangrijkste concepten in deze thesis.

Hoofdstukken 2, 3 en 4 vormen Deel 1 van deze thesis. Hoofdstuk 2 geeft een overzicht van de verscheidene formalismen voor het modelleren en simuleren van gen-netwerken, als een degelijke achtergrond voor ons werk gepresenteerd in het volgende hoofdstuk. Hoofdstuk 3 beschrijft SIM-plex, ons nieuw software-gereedschap dat een brug vormt tussen een wiskundig formalisme dat gen-netwerken modelleert, en de bioloog, die doorgaans meer expertise bezit over gen-netwerken dan de meeste wiskundigen. Het biedt

een afscherming, een toegankelijke interface naar de wiskunde, volgens een manier die het mogelijk maakt voor biologen om zelf te experimenteren met modellering en simulatie. Hoofdstuk 4 beschrijft de verscheidene toepassingen waarin SIM-plex gebruikt werd.

Het onderzoek beschreven in Deel 2 van deze thesis, hoofdstukken 5, 6 en 7, ontstond uit onze eigen behoefte voor een beter beheer van biologische informatie. Die noodzaak ondervonden we toen we met SIM-plex een groter gen-netwerk voor de celcyclus in de modelplant Arabidopsis bouwden; dit vormt een algemeen probleem in de biologie. Hoofdstuk 5 geeft een achtergrond van huidige methodes voor het oogsten van informatie uit de literatuur, maar komt tot de conclusie dat er geen geautomatiseerde of handmatige methode bestaat die voldoende potentieel toont om het grootste deel van de informatie uit de literatuur te verzamelen op een gestructureerde manier. In hoofdstuk 6 beschrijven we ons gedurfd voorstel tot een nieuwe methode om dit probleem aan te pakken: MineMap, een gemeenschaps-gebaseerd initiatief voor het manueel extraheren van informatie uit de biologische literatuur. We beschrijven de verschillende aspecten vereist om een dergelijk project te verwezenlijken, gebaseerd op onze ervaringen met ons prototypisch programma MineMap. Dit onderzoek is georganiseerd op een 'heuristische' manier, in de zin dat we een eerste schets en een werkende oplossing hebben gebouwd, die ook ervaringen genereerde leidend naar verbeteringen voor een volgend ontwerp. Terwijl hoofdstuk 6 onze nieuwe ideeën en concrete implementaties in aanzienlijk detail beschrijft, illustreert hoofdstuk 7 daarna de kern van het concept achter MineMap.

A hundred times every day I remind myself that my inner and outer life are based on the labours of other men, living and dead, and that I must exert myself in order to give in the same measure as I have received and am still receiving…
- Albert Einstein

Acknowledgements / Dankwoord

The 'Thanks' section is apparently the most viewed part of a thesis. So let me use this place to advertise that this thesis starts with an unusual but entertaining question-and-answers section (pages 7-19). It forms a light and accessible overview of my entire thesis, in the style of a magazine article interview; and so this part should be understandable for family and friends who are genuinely interested to known more about my research.

I am forever grateful to my parents, for the values they have given me and for always being there and supporting me, even when I decided to study computer science after finishing my physics studies, and also during these PhD years. I especially send many thanks to my dear mother, whose continuous energy, support and love still touch me.

Martin, thank you. You supported me, you gave me the freedom to explore and follow new routes, you guided our research to its current level, and you let me choose my own best working conditions. Our many meetings were motivating, intellectually challenging and proved to be a source of energy for my 'many many months of programming' ;) . I was lucky to work with a supervisor who could appreciate my somewhat unconventional facets, and with whom there was such a good connection on the personal level.

Also many thanks to all those that I worked with during the past years, for the interesting conversations and collaborations. Some of the important persons that helped me reach where I am today are Gerrit, Fabio, Pierre, Lieven, Kristiina, Aurine, Steven x 3, Ben, Ewa, Jens, Vladimir, Erick, Eric, Yves and Dirk, among many others. Thanks also to the people whom I had pleasant chats with 'in the corridor' :) . Finally, I won't forget the supportive crew (IT, administration, etc.) for keeping our department running. Thanks to everyone.

Greets and thanks to my group of regular friends (the food-list :)) for still coming together kind-of-weekly after all these years, and for all the laughs we share. Also greets to the various other people who cross(ed) my path, be it long or transiently. Thanks for the enjoyable social variation and for exchanging your diverse experiences and views on life.

To Ilse, my sister, and to many other members of my family: although we may see each other less frequently lately, still thanks for the beautiful memories, the time and years we spent together.

This work was financially supported by the "Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen)".

Examination committee

Prof. Dr. Ann Depicker (Chairwoman)
Faculty of Sciences, Ghent University

Prof. Dr. Martin Kuiper (Co-promotor & Supervisor)
Faculty of Sciences, Ghent University

Prof. Dr. Yves Van de Peer (Promotor)
Faculty of Sciences, Ghent University

Dr. Ir. Gerrit Beemster
Faculty of Sciences, Ghent University

Prof. Dr. Liv Thommesen
Dept. of Cancer Research and Molecular Medicine,
Norwegian University of Science and Technology (NTNU)

Prof. Dr. Martine De Cock
Dept. of Applied Mathematics & Computer Science,
Ghent University

Dr. Lieven De Veylder
Faculty of Sciences, Ghent University

Dr. Roeland Merks
Faculty of Sciences, Ghent University

Prof. Dr. Dirk Inzé
Faculty of Sciences, Ghent University

Prelude

Exploration is not a luxury. It defines us as a civilization.
It directly or indirectly benefits every member of society.
It yields an inspirational dividend whose impact on our self-image,
confidence and economic and geopolitical stature is immeasurable.
- James Cameron

Chapter 1

Introduction

1.1 Preamble

Five years of creative realization. Lively years of work, worry and growth supported by colleagues, friends and family.
Like the oral presentation that will look back on these years will need an introduction towards the less biologically adept, also this written thesis deserves a more accessibly explained first part. Also because the exploratory nature of my research lured me away from the well-trodden paths, this chapter 1 gives a descriptive look back on my past journey: an anecdotal introduction to Systems Biology for non-biologists, a draft of the road that was followed and how everything came about.
I believe that many people can be interested in the science you're doing, if you just present it to them in an accessible way, by feeding them ready pieces and making attractive, colourful analogies. In this spirit, I think a *Question-&-Answer* format, an 'interview' (section 1.3) is an excellent style for a light and down-to-earth first overview, before we go into the more serious and detailed stories in the subsequent chapters.
Also, Systems Biology is a new and emerging field where scientists usually have a background of either computer science or biology; so an accessible Q&A-introduction can reach out to both audiences. Note that the content of the Q&A section will be repeated in the subsequent chapters, although more formally and extensively.

1.2 Molecular Systems-Biology 101

In order to help biology novices better understand the accessible 'Q&A' of section 1.3, we start off with a little background information, a rehearsal of some of the basic biological concepts.

Cells
All living creatures are made of *cell*s, the basic unit of function in all organisms. For example, a tree, a mouse, and people are all built up by millions and millions of cells. In every cell, a complex biochemical factory is at work that makes cells perform specific functions. It makes a cell grow, divide, or play a specific role in the organism. For example a brain cell should perform a different task than a muscle cell. Figure 1.1 illustrates a

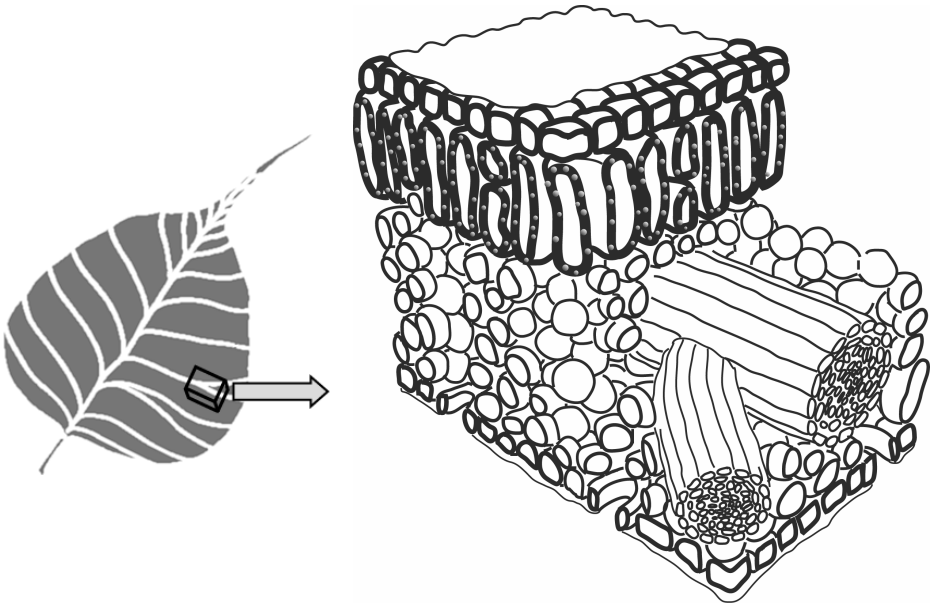


Figure 1.1. Cross-section of a plant leaf. Various cell types are visible, such as protective upper and bottom cell layers, supportive middle layers, cells forming veins to transport sugar etc, or forming (in the bottom layer) stomata (mouths) to let in air and CO_2.

cross-section of a simple leaf that contains already a plethora of cell types, each type with a different form and function.

Genes & proteins

Figure 1.2 zooms in on the *nucleus* inside a cell (top left). This cellular compartment holds the *DNA*: ultra-large molecules that store lots of information. Actually you can compare a living cell with a little computer. The DNA would be the harddisk where data and software are stored in little fragments, called *genes*. So a gene is simply one specific part of a large molecule, and it contains a piece of information.

Similar to a computer, a cell can also run programs. When that happens, some files are read from the harddisk and brought into active memory. For this, cells use a two-step process. First, the information of a number of genes is copied onto intermediary *RNA* molecules, which are sent out of the nucleus (figure 1.2, top). Second, each RNA is used to construct large numbers of identical *proteins* (figure 1.2, top right).

These protein molecules do most of the work in a cell (figure 1.2, bottom). For example some proteins work in metabolism (processing sugar, fat, minerals, etc.), and others transmit signals (like: need water, should sleep, etc.). But most importantly for this thesis, many proteins have the power to *activate* or *inhibit* (deactivate) genes back in the nucleus. As a side note, they do this by attaching themselves close to the gene's DNA sequence (on a DNA region called a *promoter*), and by recruiting a whole bunch of other proteins that together copy the gene's information again to RNA.

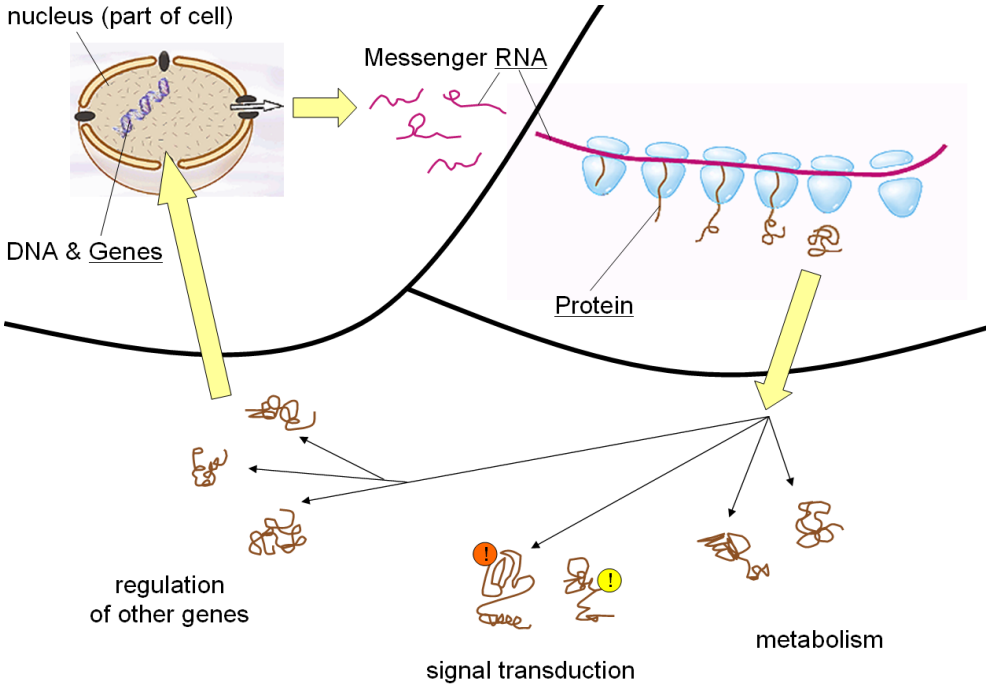


Figure 1.2. The 'central dogma' of molecular biology. *Genes* are transcribed to RNA, and RNA-information is used to construct proteins. *Proteins* are the labourers doing most of the work in a cell. Most importantly, some proteins can *regulate* the activity of other genes. Given that DNA contains tens of thousands of genes, this creates enormous possibilities for complex programs, like cell differentiation and task distribution.

Gene networks

Given that higher organisms like plants and mammals typically have tens of thousands of genes, and that many gene products (proteins) can regulate the activity of other genes, this gives enormous possibilities for complex logical wiring. When at a given time a set of genes is active, meaning that their proteins are doing some work, these proteins can determine what work in the cell needs to be done next. For example, a protein can wait until it detects a certain type of virus attack, and then start up all the genes/proteins of the cell's defence mechanism. Or starting from a little plant seed, the cell's proteins can initiate a whole developmental program; like a long cascade of decisions of what organ should be formed where, and what cell types should emerge in which tissue layer.

In summary, the combination of many gene-to-gene activations/inhibitions forms a large *gene network* (like in figure 1.3, but many times larger). This genetic regulatory network forms a structured, logical wiring that drives a cell's development, maintenance, response and other processes.

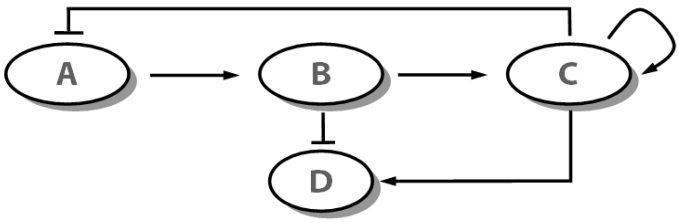


Figure 1.3. Example diagram of a simple gene regulatory network. The pointed arrows stand for activation, the flat-ended arrows for inhibition (deactivation). For example, gene A activates gene B (as explained, via an intermediate protein-step), while gene A itself is deactivated by gene C. Note that gene C can increase its own activation as soon as it becomes slightly activated.

Experimental techniques: a molecular biologist's toolbox

It is technically impossible to use a microscope and simultaneously identify thousands of different proteins in the cell, and determine what other molecules they interact with. If this were possible, we would have the ideal data to wire together gene networks and totally understand how a living creature works; and from that, we would be halfway to modify it or heal it when it's ill.

But the reality has been different for many years. An experimental, '*wet-lab*' biologist usually spends years of work to gather clues about one or two genes' role in the cell. In order to discover a gene's function or to determine protein interactions, biologists have to use techniques that often yield indirect evidence.

To name a few techniques: the most powerful (but expensive) *micro-arrays* can determine relative amounts of all the RNAs in a tissue or organ at a given time; but *relative* amounts mean that it only says that for a given gene there is a bit more RNA at time 1 than at time 2. And then there are still many obscure measurement errors. Another technique is to remove a gene (a 'knock-out'), and see if the organism loses certain functionality. After that, other techniques like Western blots, sequence analysis, yeast two-hybrid and others help to elucidate how this happens at the molecular level. In fact, yeast two-hybrid systems are also being used in an effort towards systematically testing all possible interaction combinations between two genes, e.g. (Rual 2005) in human. Still, this method has also certain limitations.

In conclusion, although new high-throughput methods are slowly opening avenues to map part of the regulatory network at a genome wide scale, wet lab biologists typically still have to work for years to definitely prove the functional relationships between a limited number of genes; or in other words, to put single 'arrows' between a limited number of genes (X activates Y, Y inhibits Z, etc.).

Molecular 'Systems Biology'

Subsequently, an integrative, '*dry-lab*' biological study brings together a larger number of gene components. Here, only the activation and inhibition arrows between genes and proteins, plus computer modelling tools, are used to justify some measured gene activity profiles over time. A molecular *systems biologist* tries to link genes together in a larger picture, as a 'system', and looks for gaps in the current knowledge. This is the challenge of the new and emerging field of molecular Systems Biology. This crucial Systems Biological goal, the validation but even more the construction of larger genetic networks, is the topic of this thesis.

All life is problem solving.
- Karl Popper

1.3 Introductory overview: an accessible, *Q&A* introduction to this thesis.

Intro **Steven, you are presenting a thesis. What is it about?**

The two main subjects of this thesis are the two projects we launched, *SIM-plex* and *MineMap*. But before I explain what those are about, let us start by saying a few words about my run-up towards them. In order to know some peculiarities of Biology, it could be interesting for our readers to hear what an initial outsider experienced when he came into the world of Biology.
Before I started my main projects, and also somewhat in during the projects, I had to learn more about biology, because my original education was in the mathematical fields. My first year in our Plant Systems Biology laboratory was more of an exploratory period for me. I dived into biology by providing a number of analytic and statistical services and solutions. I was given the opportunity to enter into cooperations with wet-lab biologists (Beemster 2005, Hilson 2004, Himanen 2004) and to apply existing software (like EASE and TMeV) to solve biological problems (Hosack 2003, Saeed 2003). Via this way I became gradually acquainted with this new world, its habits, the jargon and the 'common knowledge' of biologist that study the cells of plants and their growth.

This 'new environment' you entered, in what respect was that an adjustment?

I came from the exact (and partially applied) sciences, physics and computer science, where people speak the language of rigorous mathematics. In that world, when you want to accept a proposition, it must be accompanied by bullet-proof logical evidence. Also, in physics, measurements are usually accompanied by an exactly described margin of error. When you want to build upon the measured results, you calculate the error margins further along. And when the error margins would be comparable in size to the results, the conclusions become inexorably useless.
This stands in sharp contrast with biology, a largely non-exact science, and in its origin a descriptive science. Here, experiments are carried out on living material, with a number of replicas for which the number is often limited by financial and time reasons (organisms need time to grow). Now the significance of observations is often calculated through numerical probabilities that are in the end often evaluated with a considerable amount of 'common sense'. I have the feeling that biologists, compared to physicists, have to walk a lot more one loose ground before making a hypothesis more solid. They get, from many different sources, clues that an extremely complex biological machinery could work in a certain manner, and through experiments, they just try to get a larger certainty about it.

Clues from many different sources, you say.

Yes, that's also a huge difference with physics and mathematics. In biology one has to combine many types of information, to which often no 'unit' (kilos, metres, etc.) applies,

something that is otherwise so dear in physics. Biologists get to hear, often also from researchers in different sub-domains, that a given molecule (often a gene product) is *observed* at time X and in cell-type Y, usually with a loosely described concentration like *high* or *low*. But they can also find out that it's *involved in* this or that process, that its absence causes or *promotes* a certain *disease*, and that it, *probably*, *interacts* with some physical structure inside the cell, or that it is biochemically *active* in a given *compartment* in the cell where a vaguely defined set of possible interactors could possibly be present too.
You see, a lot of circumstantial information, which should in principle be translated back into a mathematical/physical/biochemical model. Not your average walk in the park if you ask me. Life sciences bring a large layer of extra complexity for which many exact scientists don't realize the enormous difficulty at first sight.

So, with your background in exact and computer science, what have you contributed to biology?

As a first part of my PhD project, I've looked into ways to bring exact and systematic thinking closer to the daily life of experimental biologists. This resulted in the development and several applications of the software SIM-ple*x*. The second part of my project started when we felt that not only the examination of dynamical systems was necessary, but also -if not even more a bottleneck- the gathering and management of diverse information to construct these systems in the first place. From then on, the MineMap software has been developed and applied. This is what we currently work on, and hope to continue working on after my PhD.

SIM-plex **First things first. This SIM-plex software, what's the story behind that?**

At the time, 4 or 5 years ago, our department of Plant Systems Biology was looking for ways to really bring the *Systems* viewpoint into the Biology research, and bend it more toward the exact sciences. We had a few examples from the yeast field, where research was much more advanced than in plants. Especially the Novak and Tyson example (Novak 2001) was interesting. They had studied the molecular details of the *cell cycle*; this is the cyclic, repeatable process that makes two cells from one, thus lying at the basis of growth. They zoomed in on a number of key genes that steer the consecutive phases of this process, like DNA duplication and cell division. They knew that most of the genes could positively or negatively regulate the activity of some other genes. Moreover, at the time wet-lab biologists had meticulously collected clues about which genes activated/inhibited which other genes, in a one-by-one manner. Now Novak and Tyson could build on this work and put all the information together into a so-called *gene regulatory network*.
Their success came from being able to *simulate*, or mathematically quantify, how active each of these genes is during consecutive time points in the cycle, only based on the activating or inhibiting relations between the genes. They used an extensive list of differential equations, plus many parameter estimations to get the system working in their computer program. Next, they proved it was really mirroring reality, by also simulating a few modifications: like an experimental biologist could knock out a gene to observe a change in the behaviour of an organism *in vivo*, they could silence a gene *in silico* and observe agreeing results in most cases.

So you decided you could do the same thing in the plant science.

Indeed, I thought it would be a good first incentive to link up the knowledge we had about the plant cell cycle, for it was the main research topic of our department. As I had no solid biological background, I realized I had to work closely with our biologists in order to understand and distil the needed information. Also, there I found out that some biologists also showed this desire for a more systematic approach to biological systems.
Therefore, the first thing to do was to forge a bridge between the pure mathematical language of differential equations and a biologist's preferred language in which he/she describes the logical wiring of components, like "A stimulates B under condition C". In fact, most biologists politely excuse themselves away when they are faced with a set of ordinary differential equations describing their system, let alone when they have to compile this set themselves. So we needed a lower threshold towards network building, a way to allow cooperation where a computer scientist and a biologist can sit together, or where a biologist himself can play with a system's simulations.
So we designed SIM-ple*x* (Vercruysse 2005), a simulator that tries to seek middle ground between numerical exactness of mathematics, and the fuzzy-logical world of biology where, in addition, there's even a scarcity of such exact numerical data.

You named it "SIM-ple*x*"?

Yes, this stands for "SIMulating genetic networks, with Piecewise-Linear differential Equations, in Comfortable Statements", with the ending 'cs' condensed to an 'x'.

Piecewise linear equations?...

You see, for the mathematical part of this bridging software, we used a type of differential equations that really reflect the logical nature of most of the knowledge at hand. You have to know that Glass and Kauffman had already shown years ago (Glass 1973) that transcriptional activation/inhibition of one gene under influence of one or more other genes could be approximated by a *step-function*. This means that, as soon as certain conditions are met, a gene's activation is turned on (step-up) or off (step-down). This makes that the resulting gene products will start to gradually accumulate, or to disappear. (There is also a continuous, proportional decrease caused by cell machinery that actively cleans up unused molecules). When multiple step-functions are now combined to reflect a gene's combined activation/inhibition from various origins, the gene product amount becomes a 'piecewise linear' function. The piecewise linearity is seen when the molecules accumulate or decline smoothly, until one of the activation/inhibition conditions changes and a sudden kink is seen in a graphical plot of the concentration over time (see chapter 3).

And there's still a non-mathematical part to it?

Certainly: instead of juggling with differential equations and activating/inhibiting step-up/step-down equations, in fact it's easier to just write statements in a format like "if gene A is active above a *threshold*, then it activates gene B at a certain *rate*". With the facts

presented like this, a software algorithm would have all the information that is needed to just generate the implied differential equations automatically. This is what SIM-ple*x* does. It lets a biologist enter the activation/inhibition information in this logical manner, and translates that into a set of differential equations, retranslated with every modification he/she makes. Then, it solves the equations and shows a graphical display of the simulation results.

So you made a layer of abstraction on top of the mathematics, to get closer to the biological mindset.

Indeed. Also, as a nice aside, with piecewise equations you have one numerical parameter less to estimate. Naturally you still need to say how active A should be before it regulates B (the threshold), and how much B is influenced then (the rate, up or down). But you don't need to give a gradient to the activation: instead it happens step-wise, like an immediate switch-on or switch-off.
While this and in essence the piecewise-linear framework forms an approximation of reality, and many people of pure mathematical background get the chills when they even think about approximating to that extent, one should realize that biology is a fundamentally different world. There is an incredible abundance of circumstantial data, but there is an enormous lack of precise quantitative information that would be needed to simulate exact dynamical models. This comes from the current status of experimental techniques, and should be taken into serious consideration by anyone taking the leap from the exact sciences to biology.

So your simulator offers the easy way for simulation?

Yes and no. SIM-ple*x* is meant as a heuristic discovery tool for how, or how not, genes and their proteins are linked up, and to quickly test various draft hypotheses.
It is meant to be easier, in terms of mathematics, than the full-fledged ordinary differential equation framework. But still it uses more detail than some other types of simulators. For example simulators that use boolean equations, only link up yes-es and noes instead of numbers. As a result, they derive purely qualitative answers about their models, like x 'goes up', 'stays equal' or 'goes down', which is even more crude than e.g. the experimental Western Blot measurements. While this is also useful, it is meant to answer a different scale of questions; and it usually forms a basis for elaborate analysis of all the possible logical outcomes for a given logical wiring.

But for full ordinary differential equations too, some tools exist that assist in modelling.

That's right. And that's why, after making a first bridge with SIM-ple*x*, affiliate mathematicians have now taken our first working model as a basis for further, more detailed analysis. Also, don't forget that we are working in an environment of extremely scarce quantitative data, which makes this quite a hard job to do.

Can you tell me why Novak, Tyson, you and others, would want to simulate a biological system? What is the benefit of simulating?

For that, I'd like to make an analogy. You know, the universe has spent billions of years to form and evolve the building blocks of life into the living creatures we see around us. And now we try to understand the incredible complexity of these living systems; we want to understand the interaction between the tens of thousands of finely tuned different molecular components they are built of. You can compare this, although still very simplistic, to the most sophisticated jumbo jet man has ever built, but then delivered by mail-order as just a very large bag of minuscule components you have to build it with. And they forgot to send you the manual.
So the first thing you would do, like based on the work biologists did over the past decades, is to take a close look at many components, one-by-one, sort them somehow and try to recognize some relations between them. After some time, perhaps you will figure out the concept of an engine, or a seat, or at first perhaps just a tiny little lightbulb. Then if you have on paper a draft scheme for, say an engine, the logical next step would be to actually put the components together, and see if they fit and can work together, or if something is missing, and if so, make hypotheses about how to correct your blueprint. Similarly, if you want to simulate a biological sub-system, you are first forced to think about the global picture. You have to go through the mental process of linking components together in some detail, and from the simulation results you then get feedback about the model you built. To make the analogy again: simulation lets us test-drive a system that was built from just the parts that we think we understand.

You like analogies… So to summarize, in a simulation you link up the molecular components, the genes and proteins, and see if you can create a working model about what you observe.

Yes, and it even goes further. Not only the molecular components can be linked, but also the large-scale effects they cause, the *phenotypic* traits like a plant's leaf size or biomass. I made some extensions to the basic SIM-ple*x* core to accommodate just that, based on feedback and requests I got from biologists experimenting with the software. You can read the results in (Verkest 2005) and (Beemster 2006).

Out of interest: there are many biological systems. Why would one choose to study the cell cycle?

This technique to divide a cell is a process that is shared by plants, animals and many other life forms we know. It is an interplay between genes and proteins that has been largely conserved since over a billion years of eukaryotic life. This means that although say an apple tree and your cat haven't had a common ancestor for hundreds of millions of years, still their biological tissues use much of the same basic machinery to divide a cell into two new ones, and to perpetuate life. Also cell division is one of the pillars of growth, and cell cycle regulation is tightly entangled with cell growth checkpoints. For these reasons, knowledge about the cell cycle process is invaluable for research in biomass production, pharmaceutics, cancer, and many other fields.

MineMap **Now after you made the SIM-plex simulator and applied it several times, you switched gears and launched the MineMap project. How did that come about?**

Quite naturally actually. It emerged at a time when we had some basic models about the *Arabidopsis* plant cell cycle, that were composed of molecular components that many scientists in our lab could easily sum up by heart. An objective of our group was, and still is, to extend this basic model, and to attach other, less well-known components to it, and to see to what extent we can still make it work. So the logical next step was to dive into scientific literature, and hunt for extra information. So I selected a number of review papers on the topic; reviews, since they are the most information-dense. Of course, like many other biologists do regularly, I started collecting notes while I was reading so I could look up the information more quickly afterwards. Taking notes is in fact essential. The human brain is highly overrated for its long-term capacity for remembering details, after reading a text just once or twice. Unfortunately, we forget a lot, even important details.
I kept my focus on information most useful for modelling, like genetic or protein-mediated activation and regulation. But I also valued indirect information like expression profiles, phenotypic changes and the many other flavours of information I already mentioned.

Sounds like you were facing an information management challenge.

Exactly! Quite quickly, the stack of notes I had collected had turned into a body of text as opaque as the publications themselves. Quickly finding a fact I vaguely remembered from somewhere was not much easier in my heap of notes than in the pile of papers; let alone if I had to scan through them while searching for relations and trying to see the broader picture. Moreover, I was not the only one having that problem. Working in biology, certainly in systems biology, means being confronted with so many potentially interesting facts coming from everywhere, and trying to keep a comprehensive overview.

And as a trained computer scientist, you figured you could make a computer assist you to manage the information.

Yes, I believed it should be possible to find a structured way to write down the information. Because we use a well-defined format to take notes of a publication, we can make a computer understand it, and from there do anything with it.
Fore example, one immediately useful application would be to design a software algorithm that automatically connects the loose parts that we collected, and shows them in a 'graph', a diagram that connects related biological entities with lines. Even more appealing would then be to show only a small part of this diagram, which would in its entirety be phenomenally huge, but then to make it dynamically explorable via clicking on the entities you're interested in, and make click by click more information visible. This would be like clicking on links in an internet-browser, but more graphically.
Another opportunity would be to let biologists share the fruits of their information collection efforts among each other. So you would only have to 'annotate' a few papers, but be able to browse through the information of many others as well.

Are there similarities with the iHOP software that also lets you browse somehow?

Not many. Only the browsing experience where you hop from one entity to another is similar, but then still in a totally different way. Let me sum up a few of the basic advantages of MineMap compared to this well-known iHOP software, to better illustrate the MineMap concept.
1. IHOP lets you browse only through protein names, while there is so much more to biology. I believe it's as important for making a model, that you also know something about processes, phenotypes, involved hormones etc. MineMap is designed to handle all this.
2. iHop browsing happens in a purely textual way. It shows you text snippets which have protein names highlighted as a link, so you still have to read each sentence to interpret one piece of information. But with MineMap, the human interpretation step has happened when the information was entered. So with MineMap, you're looking at digested information, and you understand its meaning at a glance. For example, a biologist will understand the kind of relation between two proteins by the type of line that connects them in the graphical diagram: an arrow is an activation, some other symbol means similarity, and so on. But the most basic difference is the next one:
3. iHOP uses computerized literature reading, or *text-mining*, to collect its results. MineMap on the other hand, relies on human-interpreted, validated information.

Then why require human interpretation? Is something wrong with automatic text-mining?

Definitely. Text-mining ís useful to get as much information from as many publications as possible, but then you get a lot of junk along with it, a lot of misinterpreted information. And especially when you try to compose a biological system's model, you don't want to build on pieces of information that were just misinterpreted by a dumb computer that in fact knows nothing about biology, or that cannot deal with the intricate complexity of human language. You will really want to rely mostly on information that is human-validated, even if you have to interpret it yourself.

How bad can it be? Is it that difficult for a computer to understand a fragment like "A stimulates B"? They must make some progress in the text-mining field, don't they?

Yes, but not enough for our purpose, or for the purposes of many other biologists. Look, there are two steps in text-mining that we need, and to make it work, they should be near-perfect. Step one is to extract all biological entity names, and step two is to understand relations between them, because that's what biologists are after.
The first step is already complicated by biologists' habits to name the molecules they study in difficult ways. That's understandable because every day, somewhere, a new gene is being studied and they have to come up with a new name. It's like if you collect cats, and a new one appears at your doorstep every day, for decades, and you have to find new names for them over and over again. In the end, you'll start naming them like the 'brown-grey-striped-blue-eyed cat with a funny taste for carrots'. And that's what biologists are doing: they make up multi-word names that describe what their protein looks like, and what other molecules it likes to stick to. But one of the problems with such names is that your colleague cannot remember the exact name and starts talking about the 'blue-eyed carrot-cat with brown-grey stripes', which is, for an ignorant computer, a totally different

name. In addition to that, it is often humorously said that biologists would rather share a toothbrush than a gene name, so your competitive colleague at the other side of the ocean will definitely find another name for it. Or, to continue with the cats, sometimes two different cats are both labelled as 'the carrot-cat' by two biologists working on different topics.
All this Babylonian confusion doesn't make it easier for computers. They have to be learned to use pattern-matching, they have to keep long lists of gene name variants, and they should be equipped with heuristic rules to disambiguate duplicate names based on the context. All these techniques are fallible because it's so difficult to cover all the many special cases. The best algorithms get a score of about 80% correctness per gene name.

And then there's step 2, extracting relationships between genes, or other biological entities. How well does that work already?

It builds upon step 1, the finding of names. As that is already far from perfect you can see that the errors quickly multiply each time an extra component enters in a relationship, and it soon becomes very unreliable. For example, take a common statement with three components: "A influences B under circumstance C". When a computer tries to recognize both the three terms A/B/C, and how they are wired together based on relation-terms like 'influences', the combined trustworthiness becomes very low.
To illustrate this, there was a competition for text-mining groups a couple of years ago, where they tried to connect genes with correct function-labels based on a fragment of text. The best scores ranged from only 1 to 10%, where the 1% score gave reliable results but missed a lot of information, and the 10% scoring results already included much false info, see a report of BioCreative: (Blaschke 2005). As you see, this is not something we can use just like that.

So apparently, computers are not smart enough yet to understand written text.

No, it's sad but computers by far do not understand the context of a biological text in all its details. It's like when you want to understand a conversation in a foreign language, and you think you'll manage by learning a few hundred words. Now you will pick up some of them, but your mind will often put them together in funny ways and essentially leave you clueless. And then still, suppose you do understand the language, then think of trying to understand what two doctors are talking about to each other, because that is the high level of language used in biological publications. That shows that you actually have to know much about biology already, before you can understand a biological text.
To give a concrete example for that, and I really like to make cat analogies, suppose I said "The cat ate the fish, and now it's dead". Of course, we all know that the fish died here. But suppose that I just said before that "The fish was poisoned". This changes everything and now we know that the cat is dead, by eating poisoned fish. If a computer didn't understand this essential contextual difference, it would think the cat is still alive and wonder why we bury it. And there are thousands (if not millions) of such things 'you just have to know'. And it's unrealistic to wait until every seemingly 'obvious' fact has been entered into a computer, before it can start to learn on its own.

The point here is: you need extensive knowledge about the world to understand a language and learn more about the world. And that is one of the main reasons why computers are so bad at interpreting complex texts.

So to get things moving, to get out of this deadlock, you say we need people to read and interpret text?

Yes, we need biologically trained people to chew a bit on the information, and then feed it to a computer in a very simple format that it can understand. It's almost like you're talking to it in an infant's language: ready pieces, simple sentences, and no ambiguity.
Only the human brain is able to decode what a human author has encoded in complex language. That is the situation today, and I predict it will stay like that for at least one or two more decades.
Still, automatic text-mining does have some merits. It can be used as a preparatory step, to find and to zoom in on information that is possibly interesting. It can scan a thousand reports and help you find many of the occurrences of for example the substance 'endorphin' and its synonyms or derivatives. But from that point on, we still need the human intellect to understand the text, or in the best case where a suggestion of an interpretation is made, we still need thorough human review.

Ok, so we need human interpretation and review. And therefore, you designed a simple language that forms the connection between man and machine.

Yes. As I said, it all started by taking notes while reading biological articles. I simplified complex sentences into their essential parts, and wrote those fragments down for future reference.
Now after you read quite some text, you start to see recurring patterns. It's like if you'd read a text about animal-lovers, you might understand that I like cats, his preferred pet is a dog, and she is crazy about dolphins. This is a pattern of information: "person X likes animal Y". You have the same in biology, where you have patterns like "protein A binds to molecule B", and "molecule C activates D under condition E", or "X is a type-Y substance", and so on. Like this, I discerned as much of the interesting information-types as possible, and I distilled the text into such simple phrases.
From the easy examples above, it may look simple to translate things into such a format. But in reality it takes some experience to cut complex biological sentences into structured pieces. If you're curious, take for example a sentence like "*Mitotic cyclin (Clb2)-bound Cdc28 (Cdk1 homolog) directly phosphorylated Swe1 and this modification served as a priming step to promote subsequent Cdc5-dependent Swe1 hyperphosphorylation and degradation*." You have six different elementary pieces of information here, which are compactly stuffed into one sentence; I refer to the MineMap chapter in this thesis, chapter 6, for a full analysis. Some sentences are easy, others are not.
It took me about ten papers, mostly 'review' papers (summarizing a number of others) about the plant cell cycle, before I could stabilise the syntax (the sentence-structure) of my elementary language for a first, usable draft. The language is designed to be easy ánd powerful enough for people to use and read, and to be understandable for computers.

Tell us more about the '*syntax*' of this new, simpler language. How do you form sentences?

I was inspired by how biologists usually write down certain knowledge. They are used to draw *interaction diagrams*, which are basically schemes of protein names with arrows or lines between them. If you have one protein stimulating another, they draw a plain arrow; if you have an inhibition, they draw a line ending with a perpendicular dash; for binding there is yet another symbol, and so on (Kohn 2001, Kitano 2005). So I started by translating the most common graphical elements to a textual equivalent, like a plain arrow becomes ->, an inhibition arrow -|, and binding <->. Also, if something happens at a certain time, location or condition 'xyz', you write '@ xyz'. Plus it draws similar inspiration from some computer languages and from mathematics. In summary, the idea is that you work with symbols that you find on your keyboard. It is in fact stenography; it's a quick-to-write language.
It's also a powerful language. Although one biologist typically won't need every aspect of it, it is designed to express many different types of knowledge that can be combined in various ways. To give more biological detail: it can cover different types of interaction, relations, value-assignments, 'time-series' (these can even be rough protein activity profiles over time, 'Western blots'), and phenotypic observations. Also vague, generalised statements like "most X's bind to Y's" are possible, or hypothetical assertions can be expressed.
What this simple language can cover today, is only limited by the types of information that I found interesting during my personal quest. Its possibilities certainly meant to be further developed and expanded based on already received and future user-feedback.

A *syntax* tells how to form sentences, but a language also needs a *vocabulary* of words. Are there rules for the vocabulary in your simplified language? How do you prevent word ambiguity?

You would see problems immediately if two biologists would put their notes together without using the same terminology. For example, one person would call a bird's feathers "feathers", while someone else would talk about "plumes", and that would lead to Babylonian confusion again. Therefore, they need to agree on a common vocabulary. They should agree to use one unique *term* for each different *meaning*.
Luckily, we have *ontologies* for that. Biomedical ontologies can be seen as lists of words with an exactly described meaning. They try to cover every possibly used term or concept in a subfield of biomedicine. Best known is the Gene Ontology that tries to cover all possible functions of genes (plus their locations and the processes they can be involved in). There's also the Plant Structure Ontology that intends to cover every possible part of every known plant, with terms ranging from large organs to microscopic compartments and biological cell layers.
Note that although a lot of ontology development work has already been done, many of the ontologies are still quite fragmented and it can occur that a term is not available yet. But this can be interesting feedback towards ontology-builders, for they can be alerted of missing terms that should be added.

Alright, we've talked about scientific articles, biologists reading them, and human-translated excerpts in a simple format that computers can understand. What's next?

After the design phase, I programmed the *MineMap* software; it's in fact a pipeline of algorithms. The first part analyzes and recognizes the elementary extracted pieces of information that I talked about, and translates them into an internal data format. The second part then composes a list of all the relations that are presentable in a diagram. The final part serves as a first demonstration of what you can do with the collected information. It is an interactive visualiser that lets a user select which part of the relation-information he wants to see, and lets him browse and explore it.

Can you explain *MineMap* in simpler terms?

Basically, I've written software that reads a text with simple information like "A stimulates B", and then shows the information in a picture with an arrow from A to B.
Moreover, if much more is known about B, then you can double-click on B, and a lot of extra arrows and lines jump out, towards C's, D's, E's and what have you.
It combines all the loose pieces of information coming from various sources, and puts them together into a huge interconnected diagram, of which only part is shown at a time. *MineMap* supports biologists to '*mine*' information, and to '*map*' it in a diagram.

It sounds a bit like a 'mind map'-type diagram. But then an interactive one.

Indeed, the name '*MineMap*' is a pun on the term '*mind map*'. A mind map diagram is for people a very natural way to present related concepts, since it is similar to how our brains work. Look, if you think about some thing, then many related areas in your brain also become activated. In an example like Proust's "La Madeleine" (Proust 1913): if you were told to think about the concept 'childhood', immediately some memories should pop up. Perhaps you think about playing in a garden, or the long summer vacations, or your first dog. And from your first dog, perhaps you think about the colour of his coat, and from that, how wet it was after a swim, and how he used to splatter you all, and so on.
This concept-hopping is the way how we people explore our memories. And to have the tremendous amount of biological information available like this is a very interesting and intuitive opportunity.

So it looks. But first, many people should bring pieces of information together. How can biologists share their individual information?

For that, I built a website where biologists can enter information they extracted from publications. The information is stored in a database and every visitor on the site can look at these extracts, or they can immediately browse the combined information in the 'mind map' representation.
When someone adds information, the information is immediately shared and visible for everyone. The website is still in its infancy, but you might already draw parallels with the concept of Wikipedia, the free online encyclopaedia that is built and edited by visitors like you and me. In fact, we are now trying to bring this cooperative *Web 2.0* concept to Biology and Biomedicine.

What is this "Web 2.0" ?

This is nowadays the "Big Thing" on the internet: many websites are now based on *user-contribution*. This means that website creators rely on their visitors to add new information and value to their site, to build it up and give it content. And all visitors benefit from each other's contributions. When it works, you get a result that no small group of persons could ever achieve on its own. People now work like a colony of ants. They all do a little work, they build something great, together, and they all benefit from the big result.
I think here in Biology, we can learn from successful examples of 'social' sites like Wikipedia, MySpace, Facebook, CouchSurfing, and many others. These *Web 2.0* websites often form a *social network*. They have attracted a large, loyal base of users that keep returning and help building content, purely out of enthusiasm. And they welcome many new contributors every single day.
The *Web 2.0* principle is: "The more people contribute, the more value is created, the more curious visitors will become, and the more these new people will contribute again." This is a self-fuelling process, and all you need is a bright idea, plus some critical mass to get things booming.

Won't you need some precautions if just anyone can change the content?

Absolutely. Now, our prototype web service is still small. We know all the people who use it, so changes they make are still manageable. But when the site and its user base will grow larger, then we'll definitely have to take measures. We'll have to deal with community building challenges. Just like a young company that hires more and more new employees, we'll need to make some hierarchy for user reliability and responsibility (like administrators, experienced users, novice guests, etc.).
Also, just like Wikipedia, we should install bots (robots) that detect vandalism. Just try to delete a whole Wikipedia article (no, don't), and within minutes it will be resurrected (and you will be banned?). This is possible because all changes are logged, so it's easy to reverse them. MineMap already logs changes too, but a lot of work can still be done towards the future.

Ah, so MineMap is still a prototype that should be further developed?

Yes. This is a large project that shouldn't end with my PhD defence. For my PhD, I present the following achievements: (1) the design of the whole MineMap concept and workflow, (2) the demonstration of a first working prototype, (3) its first applications, and (4) based on feedback and new insights, requirements to take future steps towards a large-scale web-application.

And these insights for future directions form an essential part of the research.

We have received a lot of feedback since the first version of the prototype. Several people have volunteered to get hands-on experience with the software, and a number of valuable updates have been implemented based on their suggestions. But now the time has come to

write a PhD thesis, as an intermediate milestone. MineMap is a large project-in-progress, so this thesis should include a look back, as well as a look forward.
To name a few things: there are concrete ideas for vocabulary extension, which should also cascade through all parts of the pipeline and the interactive browser. Next, even though this visualiser is powerful and visually very attractive as it is, there is still room for many extensions like complex filters to specify even better what people really want to see. We are currently setting up collaborations to help us with that part. Also, now we have a website based on a Java-applet, but a solid and attractive Web 2.0 site must at least partially use something like the powerful PHP web-application framework.
To recapitulate: before we take a long run towards the next, distant milestone, as part of this thesis it is instructive to have guidelines for future requirements, to have a directions for prosperous continuation based on our current insights.

Part 1 -
Gene Network Analysis

All models are wrong,
but some are useful.
- George E. P. Box

Chapter 2

Modelling and simulation of biomolecular networks

In this chapter we review the motivation and art of building and validating biological models. We take off with a general introduction to modelling in section 2.1. After that, in section 2.2, we illustrate the broad spectrum of modelling and simulation formalisms, each solving different biological questions.

2.1 Biological modelling and simulation

Biomolecular networks at the basis of life
At the basis of all living creatures, in the make-up of all their cells, lies a complex molecular machinery consisting of thousands of different interacting biomolecules. These components, be it genes (DNA), RNA, proteins or smaller molecules, form a highly sophisticated network of interactors. They respond to external events like virus attacks, sunlight or cold, and to internal signals like cues for growth, nutrient need, or cell differentiation in developmental programs. Signals that arise are usually transmitted through various parts of the network, via different genes being activated, biochemical processes being turned on or off, and generally can result in a whole new biochemical configuration being attained. Each of these microscopic, finely tuned biochemical network states can have an effect on the macroscopic functionality that enables an organism to grow, develop and survive in its environment.

These networks are essentially formed by genetic interactions
In order to understand the biochemical clock-work that makes a living cell function, biologists often start by studying the central players in this network: the genes. Gene products, like proteins, can bind and interact with a limited number of other molecules, which can lead to a structural modification, for example the addition of an extra phosphor group. A modified protein usually attains a different functional state: very often this state changes between *active* or *inactive*. Activity means the possibility to perform certain actions, like binding to again a number of other molecules, or participation in certain biochemical reactions. Most importantly, through such biochemical interactions, many gene products can even switch on or off other genes.

(a) Figuring out how the individual parts make up the car: modelling

This regulation of gene activity, or gene expression, is usually achieved through the cooperative influence of several other gene products. The connected interactions between DNA, RNA, proteins and other biomolecules back to DNA, form logical structures like feed-forward, positive feedback and negative feedback loops, whose function is to process and transmit the parallel signals coming from various sources (see figure 2.1 and 1.3). The first task of a computational biologist is then to put together a large number of these genetic components and compose a logical *model*, an outline of the structured circuitry.

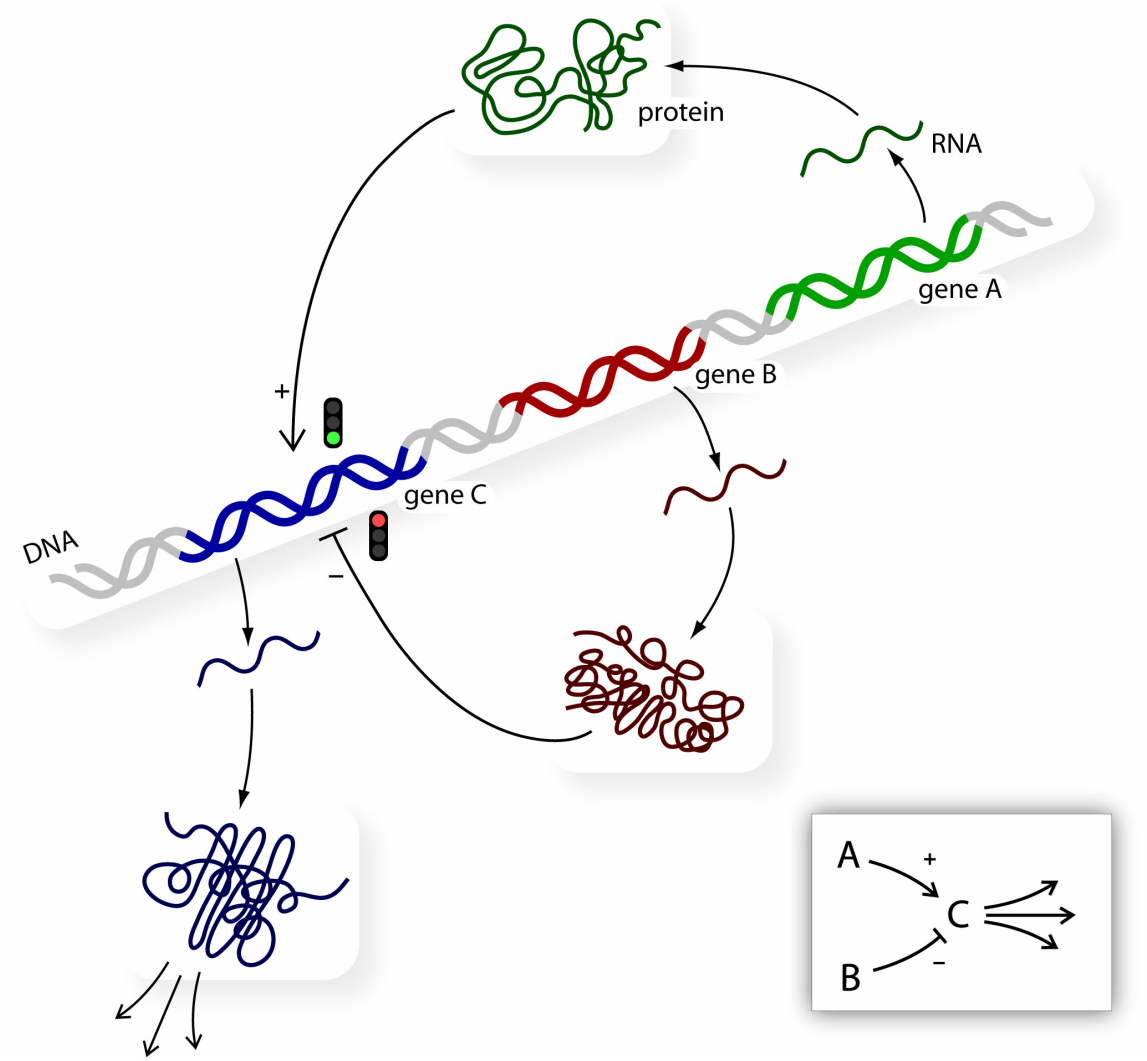


Figure 2.1. A genetic network fragment. Genes A and B, located on the DNA strand, both have a regulating effect on a third gene, C. Typical regulations are activation and inhibition, but could also include enhancement, attenuation or coregulation. Regulation happens via intermediate biochemical steps of DNA transcription to RNA, and RNA translation to proteins, but it is often simplified to a logical model (bottom right).

(b) Driving that car: simulation

But with a model alone, it can be hard to comprehend how the genetic network really works, what its temporal dynamicity is, just by intuition. Therefore, the indispensable next step after drawing the model is to use mathematical methods and computer tools to analyse the model. One can explore how the different components behave and interact over time, and what would happen if some components or interactions would be modified. This *in-silico* exploration of the temporal behaviour is called *simulation*, and SIM-plex

(Vercruysse 2005) is one of the tools that enable this. SIM-plex is only using one of a large number of simulation methods. There are a whole range of modelling & simulation techniques being investigated, from fine-grained to coarse approaches. Each of these methods has its own advantages and issues, but may be appropriate to answer only a certain range of biological questions.

Scope of the techniques

Over the past decades, biologists have been collecting evidence for many one-to-one gene interactions. More recently, powerful experimental screening techniques like microarrays are also providing a view on the entire spatiotemporal gene activity footprint of cells. Also more recently, bioinformaticians are analyzing the genetic sequence and generate clues about regulatory sites, for example (Rombauts 2003) or (Davuluri 2003). Combined, a wide range of interesting information becomes available to build genetic networks of considerable size, and this for more and more different species and biological processes.
While this is sure promising, much information is still missing (like many components, interaction details, or quantitative data), so the road towards a complete 'virtual cell' is still a long one. Also, the computational cost makes it prohibitively hard to reliably analyse the cooperation of too many genes at once. Luckily, while the whole genome of an organism typically includes tens of thousands of genes, it was observed that genes can usually be grouped into functionally related modules that are relatively independent (Hartwell 1999). This does makes it possible to perform computational analyses of average to high detail, if we limit our study to only tiny fractions of the vast biological network. Typically, such modelling and simulation efforts only deal with a manageable ten to thirty components at a time. In addition, the human comprehension factor also plays a role. Especially user-friendly tools that support modelling and simulation will allow more complex and large genetic networks to be explored and analyzed.

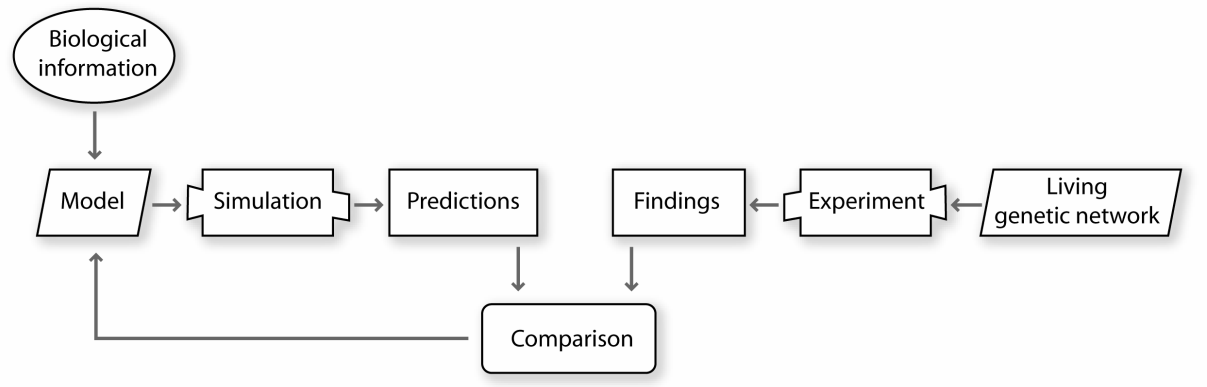


Figure 2.2. Basic loop of modelling, simulation and validation. This includes composing, simulating and adjusting a mathematical model, based on a priori knowledge and comparison of predicated information with new experimental findings.

The modelling loop; applications

Ideally, genetic regulatory systems can be analyzed via the combined application of wet-lab experiments and dry-lab computational tools, see figure 2.2. Starting from biological information coming from literature and data coming from databases and biological experiments, a model can be put together in a mathematical framework of choice. From there, a number of simulations are to be executed based on a range of initial conditions

and on a number of small modifications of the model network. The simulation results are then compared with existing or new experimental findings, like RNA expression profiles or protein activity Western Blot time-series. Discrepancies between the two can suggest, if the experimental data is reliable, that the model's fine-tuning (parameters) or its structure (included components and interactions) should be revised. Ultimately, this process makes us understand how particular complex patterns of behaviour emerge from regulatory networks, based on the interaction between genes.
Applied to model organisms, this means that we look for genes and interactors that make the organism grow, develop, differ from other species, and respond to their environment. This understanding of life at the basic biomolecular level constitutes a huge scientific challenge with potentially high industrial pay-offs, by fuelling hypothesis-driven design and biotechnological engineering.

2.2 Overview of modelling & simulation formalisms

To draw the context wherein the SIM-plex simulation software finds its place, we now present an overview of the wide spectrum of mathematical modelling and simulation techniques (de Jong 2002, Gibson 2001, Ideker 2003, Li 2008, Smolen 2000). We summarize some of the most important mathematical formalisms to describe genetic regulatory networks, and outline their application possibilities, limitations, and intended purposes. Still, the list of modelling techniques described here is not meant to be exhaustive.

2.2.1 Directed graphs

The most common way to represent a genetic network is as a *directed graph*. This is a graphical representation with nodes that represent genes, proteins or metabolites; and directed edges that connect the nodes, representing the biochemical interactions. There are typically two kinds of edges, one type saying that it is activating the target node (label "+", or a "--->" arrow), while the other deactivates the target node (label "-", or a "---|" arrow).

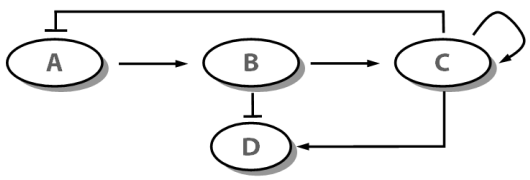


Figure 2.3. Simple gene regulatory network (from chapter 1), represented as a directed graph with two types of edges. The pointed arrows stand for activation, the flat-ended arrows for inhibition.

Representation scope

The directed graph example of figure 2.3 represents a gene regulatory network. It displays two genes A and B that activate another gene, a gene C that activates itself (usually as soon as it becomes activated by B), and two genes B and C that inhibit the production of another gene. Alternatively, this very same representation could be used for mixed component types. Suppose that D is not a gene but a metabolite. Then (as on possible interpretation) this figure could tell us that in fact the process that produces B is inhibited, while the gene-product C could for example be much closer involved in the production of D. In any case, these models allow for a significant amount of ambiguity, or abstraction, depending on the message they are meant to transmit. As such, they form a widely used heuristic approach to think and discuss about biological networks.

Network construction

Larger networks can be manually or automatically inferred from fragmentary knowledge in literature and databases, or reverse engineered from expression experiments, like with Bayesian networks (Michoel 2007). Also, one often clusters microarray gene expression profiles via a similarity metric (D'haeseleer 2000); similarities in expression can be taken as connections between genes, even though they are at first undirected. Also, bioinformatics techniques can suggest a list of electronically inferred regulatory relations based on transcription factor and regulatory motif relations. These larger networks typically comprise hundreds or thousands of genes.

Network analysis
In a next step, one can carry out an analysis of the network structure to investigate behaviour based on structure (e.g. feedback loops, see the ODE section below), and from there make structural hypotheses based on observed behaviour. Also, one can study network topology, like global connectivity and recurring patterns (Schlitt 2007). Interestingly, genetic networks appear to be highly modular (Hartwell 1999, Thieffry 1999). This is an important point for the field of modelling and simulation, since as a consequence the large genetic networks can be decomposed and studied via smaller, relatively independent modules.

More detailed graphical representations
These simple directed graphs can be extended with many more node and arrow types. They can even use arrows that connect more than two components. Like this, much more detailed biological and biochemical details can be conveyed. Well-known representations of this type are the *Kohn diagrams* (or *molecular interaction maps*) (Kohn 1998, Kohn 2001, Kohn 2005, Kohn 2006) and the Kitano graphical language (Kitano 2002, Kitano 2005). For more information about these detailed graphical representations, we refer to the MineMap chapter.

2.2.2 Bayesian networks

In the Bayesian network formalism, a genetic network is represented as a probabilistic, directed *acyclic* graph, meaning that no path from any node leads back to itself, see figure 2.4. This constraint is a requirement for the mathematics involved. Each node represents a gene and captures its expression level, and each arrow stands for a supposed conditional influence between genes' expressions. An arrow's weight indicates the size of the influence, and the goal of *learning* a Bayesian network structure is to find the optimal weights. For each node or gene in the network, a conditional distribution function is also defined: $p(X_i \mid parents(X_i))$. This is a mathematical probability function that describes what the gene *i*'s expression X_i will be, given the expression of all its parents in the network, meaning all the nodes connected via an inbound edge.

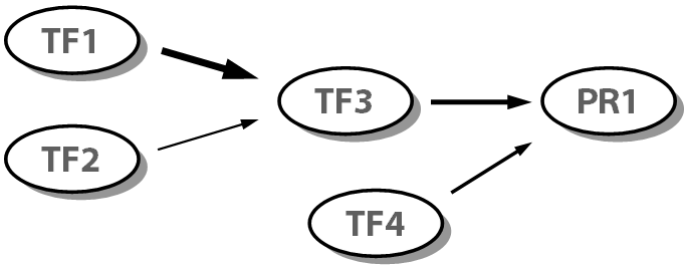


Figure 2.4. A Bayesian network. A directed acyclic graph with nodes that represent genes, and weighted arrows that stand for supposed interactions, with a size that depicts the influence of the parent on the child gene.

Learning structure and gene relations
While the mathematics of the joint probability distribution of the entire graph can be quite daunting at first, it can be considerably simplified by taking into account conditional independencies between unrelated genes. This has cleared the road towards learning techniques that find an optimal structure for the network. Such an optimal solution would

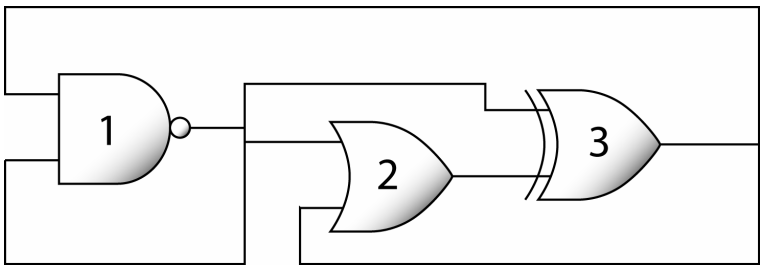
consist of the best estimates for the conditional parent-child dependencies, as derived from a preferably large set of expression profiles for the genes in the network (Friedman 2000, Ong 2002, Pe'er 2001, Perrin 2003). Proposed solutions will usually be suboptimal, as the available data (typically only a few dozens of expression snapshots) severely under-defines a network of thousands of genes.

Discussion
Since Bayesian networks have a solid basis in statistics, they are capable to deal naturally with noisy results coming from measurement errors and the stochastic nature of gene expression. They are also useful when dealing with incomplete knowledge of a system. On the other hand, the acyclic requirement of the basic Bayesian framework is an oversimplification. This is especially true for our research department's main investigation topic, the cell cycle, which is a long cascading cycle of re-occurring, regulating interactions. To some extent, extensions to this basic framework have been proposed that deal with feedback relations (Kim 2003).

The representations discussed until now only serve as topological modelling tools, to investigate the biological network structure and logic. In the following sections, however, we address formalisms that also explicitly deal with the dynamical nature of gene regulatory networks, in order to study behaviour and evolution over time.

2.2.3 Boolean networks

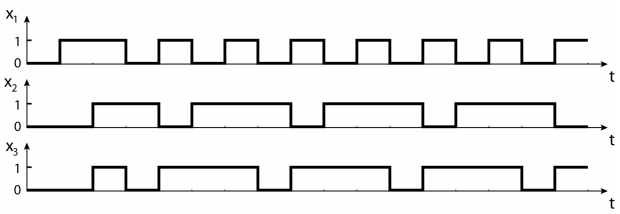
The most elementary way to describe a gene's activation state, is simplifying it to just *on* (active, 1) or *off* (inactive, 0). When it is active, its gene products are supposed to be fully present, when inactive, they are totally absent. Gene interactions are then described in the framework of Boolean functions, for example: if gene x_1 is 1 AND gene x_8 is NOT 1, then gene x_3 will become 1 at the next time-point. Figure 2.5 illustrates a Boolean system.



(a)

$x_1(t+1) = x_1(t)$ nand $x_3(t)$
$x_2(t+1) = x_1(t)$ or $x_3(t)$
$x_3(t+1) = x_1(t)$ xor $x_2(t)$

(b)



(c)

Figure 2.5. Example of a Boolean network, equations and simulation. (a) The logical ports in the Boolean network denote the (in this case binary) boolean functions assigned to the genes' activation. The different logical symbols are NAND/OR/XOR ports. (b) The time-dependent equations, determining the gene's state based on its logical port's input given by other genes. (c) would be a simulation of the gene activity dynamics over time, which shows a cyclic behaviour for this toy network.

Trajectories and attractors
At every discrete time-step, the system of *n* genes is in a certain state, given by *n* true/false values (a *state vector*). A valid transition is made if two states are connected by an application of the boolean functions. A sequence of such transitions forms a *trajectory* through state space. As the number of states is finite, the number of trajectories is finite, and lends itself to systematic analysis. One can search for steady states or state cycles, also called as point *attractors* or dynamic attractors.

Application
An interesting application of Boolean networks is in the study of large-scale regulatory network behaviour. For example, analysis of networks of up to 10000 elements showed that, when the number of regulators per gene is not too high, the expected median number of attractors is proportional to the square root of the number of genes. This means that a system of 10000 genes would have an expected 100 different attractor cycles or states (de Jong 2003, Kauffman 1969, Kauffman 1991). One can see an attractor as the genetic expression profile of a certain cell type, and therefore as one of the possible end states of a developmental differentiation process. Kauffman argued that this seems in accordance with the observation that the number of cell types seems to grow with the square root of the number of genes in an organism.

Considerations
While Boolean networks allow large genetic systems to be analyzed in an efficient way, they form a coarse simplification of biomolecular reality. For example, abundance effects of gene expression like experimentally verified in (Verkest 2005), can not be simplified to a simple on/off state. Also, state transitions of a group of genes do not happen in a synchronous manner in reality. As a consequence, certain behaviours may not be predicted correctly by the Boolean framework, and this requires more general methods.

2.2.4 Generalized logical networks

As a generalized form of the Boolean mechanism, this framework allows the state variables to have more than two discrete values, as described in (Devloo 2003, Thomas 2001). The real concentration x_i of a gene product *i* is now mapped onto a discretised abstraction x'_i , based on a number of threshold concentrations. These are thresholds of this gene's influence on other elements of the regulatory system, and they support the abundance effect mentioned before. If a gene influences *k* other genes in the network, it may have *k* distinct thresholds. Like this, the component abundance space, or *phase space*, is divided into boxes separated by these thresholds, much like in the PLDE formalism (see section 2.2.6).

Application
This logical method has been demonstrated for example on modellers' pet genetic networks like the small regulatory network of λ phage infection in *E. coli* (Thieffry 1995) and pattern formation genes in *Drosophila* (Sánchez 2001, Sánchez 2003). Also, (Mendoza 1999) used it to study flower morphogenesis control in *Arabidopsis*. They built a model by

taking information from several publications with genetic and molecular data, they distilled a small genetic network from that, and they chose fitting logical functions. This resulted in a gene regulatory network with a number of steady states corresponding to the different gene expression patterns in the floral organs of plants (sepals, petals, stamens and carpels).

2.2.5 Ordinary differential equations (ODEs)

As one of the most widely known formalisms for modelling genetic networks, *ordinary differential equations* are used with the aim to perform rather detailed simulations that yield quantitative *in-silico* data. They use real-number, non-negative variables x_i to represent the amount or the concentration of the gene products RNAs and proteins, any kind of other molecules, or molecular assemblies in the cell. Regulatory and other interactions between them are modelled by differential equations, as in: $dx_i/dt = f_i(x_1, \ldots, x_n)$, $1 \le i \le n$, with n the number of components, and $f_i : R^n \rightarrow R$ often nonlinear functions. Even discrete time-delays τ_{ij} due to a lag in transcription or translation completion can be modelled, via a straightforward adaptation like: $dx_i/dt = f_i(x_1(t-\tau_{i1}), \ldots, x_n(t-\tau_{in}))$, or via integrals that deal with distributed delays. For modelling specific biochemical reactions with the functions f_i, several methods exist, like the Michaelis-Menten equations describing the kinetics of many enzymes (Michaelis 1913, Briggs 1925).

Common simplifications

Sometimes a simplified form of ODE equations is used, abstracting each component's regulation to a summation of distinct influences from other components. The equations then reduce to: $dx_i/dt = \Sigma_j \kappa_{ij} r(x_j) - \gamma_i x_i$, where κ_{ij} and γ_i are creation and degradation rates, respectively. They form a balance between creation by gene transcription, RNA translation or protein activation, and clearance caused by molecular degradation, destruction, diffusion, and cell growth dilution.

The production term shows a nonlinear regulation function r. When it comes to RNA-translation, one can usually take a linear relation proportional to x_j. But for regulatory influences working on gene activation, one often uses a nonlinear regulation function like the Hill equation: $r = h^+(x_j, \theta_j, m) = x_j^m / (x_j^m + \theta_j^m)$. Figure 2.6 illustrates how this equation shows a sigmoidal shape, which is in agreement with experimental evidence (Yagil 1971, 1975). It approximates the gradual activation of one gene under the quantitative influence of another component. The parameter θ_j is the activation threshold; in a logical framework this could be interpreted as where the target gene becomes activated sufficiently under the influence of the activator, to enter the 'on' state. The parameter m determines the steepness of the sigmoidal shape: the larger m, the steeper the transition zone from almost inactive to nearly fully activated. Also, for gene-deactivating influences, the inverse of h^+ is used: $h^- = 1 - h^+$; this is a mirrored sigmoid that goes from high to low activation based on the inhibitory component's abundance.

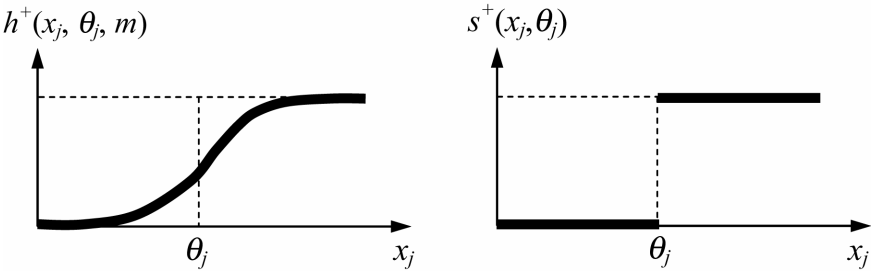


Figure 2.6. Examples of nonlinear regulation functions, used as gene-gene interaction modelling functions in the Ordinary Differential Equations. (a) The Hill function h^+ (discussed in section 2.2.5), and (b) the Heaviside function, or step function (discussed in section 2.2.6).

Analysis

The nonlinearity of the ODEs' f_i-s makes it in general impossible to analytically solve the equations. Only in special cases, steady states and attractors can be analysed, in order to investigate the effect of interaction feedback loops on the dynamical properties of the system (Smolen 2000b, Tyson 1978). These feedback loops in the network structure make two remarkable behaviours possible. A negative feedback loop, like in figure 2.3 (top loop), has an uneven number of inhibitory influences, and may cause the system to reach or oscillate around a single steady state. A positive feedback loop has no or an even number of inhibitory interactions, and makes the system end up in one of two possible stable states, depending on the system's initial state. These systemic behaviours have immediate repercussions on biological networks, where they give rise to cyclic behaviour such as the cell division cycle or control of molecular concentration stability (homeostasis), or where they cause bifurcative choices between end-states, like the diversification of cell types in developmental control programs.

Simulation

Next to simplifying the models, analyzing the nonlinear complexity can also be tackled via *numerical simulation*. Via this technique, an exact solution of the differential equations is approximated by starting from an initial state, and calculating approximate values x_i for a whole interval of consecutive, closely spaced time points. Numerical simulation has been enabled by simulation software tools, like there are GEPASI (Mendes 1993), DBsolve (Goryanin 1999) or SOSlib (Machné 2006), and has been applied to a range of previously well-studied regulatory networks, like the λ-phage growth-control switch circuitry (MacAdams 1995), *lac*-operon induction in *E. coli* (Yildirim 2003), or circadian rhythm control in several organisms (Goldbeter 2002, Ruoff 2001, Ueda 2001). Simulation can be followed by a bifurcation analysis, where one studies the outcome of different initial states and steady state and attractor stability. Like this, Novak, Tyson et al. have analyzed different cell cycle networks, focusing on gene regulation and post-translational modification (protein phosphorylation). For example they applied ODE simulation and cell cycle model analysis to frog eggs (Novak 1993), to fission yeast (Novak 2001), to budding yeast (Chen 2004), and to the mammalian cell cycle (Novak 2004), and they analyzed a generic model for the cell cycle (Csikász-Nagy 2006).

Limitations

Although one may be idealistic and believe that one can solve everything with just ODE's, there are some practical hurdles that hamper their wide-spread application. As they form

a precise numerical technique, they require considerable quantitative measurements of the kinetic parameters in the equations, and the experiments to gather such numerical data have not or cannot be performed. Therefore, they are often hard to use for the many less-quantitated biological systems. For example in the cell cycle networks mentioned above (like Novak's), most of the abundance parameters had to be chosen in a trial-and-error way, so as to replicate a qualitative behaviour of the system.

2.2.6 Piecewise-linear differential equations (PLDEs)

A further simplification of the ODE differential equations consists of abstracting away from the biochemical details of gene activation. The gradually activating sigmoid curves of the previous section are now replaced by switch-like activation steps or Heaviside step-functions, see figure 2.6. This abstraction may seem coarse, but in fact it was shown already by Glass and Kaufman (Glass 1973) that this has little or no effect on the qualitative behaviour of the system, certainly when a gene is regulated cooperatively, by a number of regulators, as is often the case. The choice for this simplification is further justified by the fact that exact quantitative data is not available anyway.
As a result, the differential equations become reduced between each activation or deactivation threshold. Within the boundaries of each threshold-separated hypercube in component abundance space, they have no nonlinearity anymore, as there, their activation regulation becomes constant and independent of the other components. In each hypercube, there will be a slightly different set of linear differential equations. This forms a piecewise-linear differential equation (PLDE) model.
When including simple degradation regulations like the constant γ_i-s in the ODE-section, the piecewise-linear equations can be defined as:

$$dx_i/dt = \Sigma_{j\in L}\ \kappa_k s_k(x_j, \theta_{jm}) - \gamma_i x_i \ ,$$

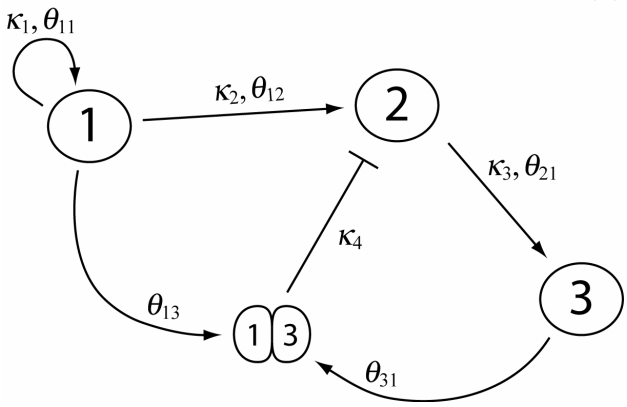
where $x_i \geq 0$ is the cellular concentration of gene product i, the $\kappa_k > 0$ are rate parameter constants, and the γ_i are degradation rates. L is the set of indices of components that influence component i, and the functions s_k are the activating or deactivating step-functions s^+ or s^-, defined as:

$$s^+(x_j, \theta_{jm}) = 0 \text{ for } x_j < \theta_{jm}, \quad \text{but 1 for } x_j > \theta_{jm}; \quad \text{and} \quad s^-(x_j, \theta_{jm}) = -\, s^+(x_j, \theta_{jm}) \ .$$

Here, the θ_{jm} are the thresholds for a switch-like activation or deactivation step; e.g. θ_{12} would be the 2nd threshold in component 1's activity. Piecewise-linear differential equations have been studied to considerable extent in the computational biology field, e.g. (Batt 2005, Casey 2006, de Jong 2003, de Jong 2004, Edwards 2007, Glass 1973).

Example model
As an example, figure 2.7a,b shows a concrete regulatory network with corresponding PLDEs. This genetic network fragment is driven by an auto-activating gene: as soon as its protein 1 is present above an amount/concentration θ_{11} (this is an initial condition, arisen by undefined causes), then protein 1 is synthesized at rate κ_1. Next, this gene 1 also activates gene 2, as soon as its protein 1 is present at a concentration higher than a threshold θ_{12}. Likewise, protein 2 will activate gene 3. And finally, as soon as both components 1 and 3 are synthesized and active above their highest thresholds, they will bind to form a heterodimer, so that they together deactivate gene 2. This deactivation can be an attenuating effect, or a complete blocking of any other influences that try to

activate it. In that case, it could block against gene 1's activating effect. All three components are subjected to a degradation rate proportional to their abundance, which is reflected by the $-\gamma_i x_i$ terms in the equations.



(a) **(b)**

$$dx_1/dt = \kappa_1 s^+(x_1, \theta_{11}) - \gamma_1 x_1$$
$$dx_2/dt = \kappa_2 s^+(x_1, \theta_{12}) - \kappa_4 s^+(x_1, \theta_{13})\ s^+(x_3, \theta_{31}) - \gamma_2 x_2$$
$$dx_3/dt = \kappa_3 s^+(x_2, \theta_{21}) - \gamma_3 x_3$$

(c)

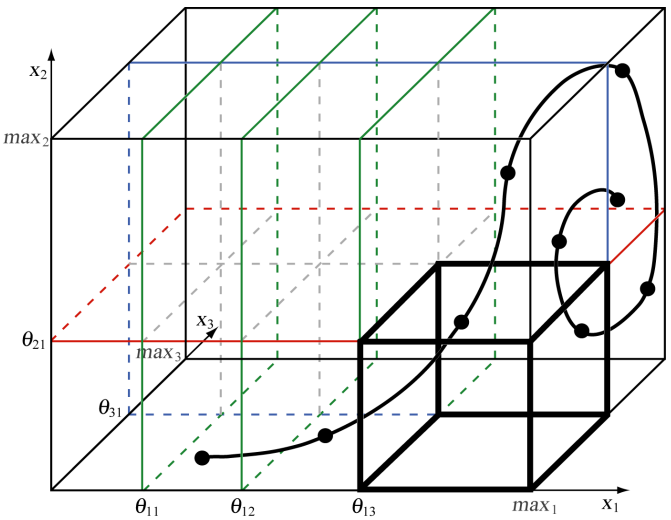
```
if x1>10 then x1  5
if x1>20 then x2  5
if x2>10 then x3  5
if x1>30 and x3>10 then x2 -8
```

Figure 2.7. Example network illustrating Piecewise-Linear Differential Equations (PLDEs). (a) Example of a regulatory network with applicable parameters, used in: (b) the corresponding piecewise-linear differential equations. (c) is a concrete SIM-plex network definition for that model (see Chapter 3).

Connection to SIM-plex and logical models

Figure 2.7c shows a formulation in logical statements of the differential equation's activating and deactivating influences. Statements like this can be entered into the *SIM-plex* software, which we developed during our research (Vercruysse 2005) (see later section), and which translates and combines those statements into the mathematical language of PLDEs. For example, the first statement reads as 'if component x_1 rises above a threshold of 10, then the creation rate of x_1 is increased with 5 units per time-unit'. This formulation hints at the relation between quantitative models in PLDEs and the qualitative logical formalisms discussed earlier.

(a) **(b)**



In the highlighted box:

$$dx_1/dt = \kappa_1 - \gamma_1 x_1$$
$$dx_2/dt = \kappa_2 - \gamma_2 x_2$$
$$dx_3/dt = - \gamma_3 x_3$$

Figure 2.8. Phase space plot of a PLDE model. (a) The phase space of the model in Figure 2.7. The threshold planes divide the space in 16 compartments where the differential equations reduce to a linear set. Illustrated is a trajectory through this space formed by the varying abundances of the components as dictated by the differential equations. The example system will end up in a cycle spiralling around and towards $(max_1, \theta_{21}, \theta_{31})$. As an example, (b) shows the linear equations for the highlighted compartment.

Example phase space plot

The resulting dynamical behaviour of this example network can be plotted in phase space, in a three-dimensional component concentration box: see figure 2.8a. This box is divided into compartments (or *domains*) by planes that represent concentration thresholds for each of the three components. In each domain, the production rate part of the PLDE equations reduces to a constant μ_i: $dx_i/dt = \mu_i - \gamma_i x_i$. In our case, we have (3+1)x(1+1)x(1+1) = 16 domains. As an example, one domain is highlighted: $\theta_{13} \le x_1 < \max_1$, $0 \le x_2 < \theta_{21}$, and $0 \le x_3 < \theta_{31}$. Figure 2.8b shows how the step-functions, which now reduce to constants 0 and 1, make the equations become linear in each such domain. E.g. the PLDE for x_1 reduces to an equation in which each term is either a constant, or the product of a constant times the first power of x_1. In addition, the equations are orthogonal, as each equation has no more terms depending on any of the two other variables.

Focal states of domains

The special form of the equations inside each domain makes mathematical analysis considerably easier. In each domain, the trajectory of the system tends to evolve towards a steady state $x_i^* = \mu_i / \gamma_i$ (i=1,…,*n*) , called the *focal state*. This focal state may lie inside or outside that domain, see figure 2.9a,b. In the latter case, the state of the network will evolve towards one of threshold planes bounding the domain. When a threshold plane is crossed and a new domain reached, the system may evolve towards a new focal state. Whether a threshold plane can be crossed or how it is crossed, depends on how the PLDEs are defined in that threshold plane. Because the step-functions are discontinuous there, the mathematical details can be quite complex. Several alternatives have been studied to deal with these lower-dimensional threshold planes/lines/… and to calculate their focal states (de Jong 2004, Casey 2006, Gouzé 2003).

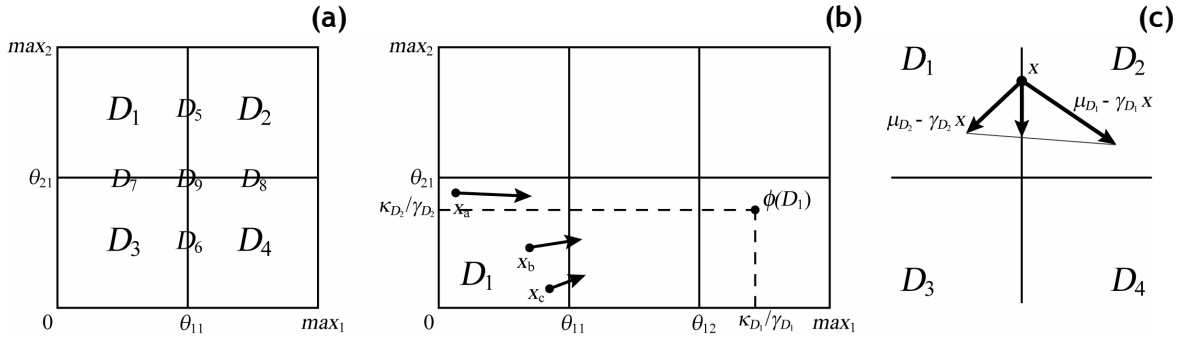


Figure 2.9. Logical domains in the analysis of a PLDE model. (a) shows the phase space plot of a simple two-dimensional system, indicating 4 regular domains and 5 switching domains. (b) Inside a domain, any state of the system tends to evolve towards a focal state $\phi(D_i)$. (c) A way to define gliding behaviour on a one-dimensional threshold plane between domains that push towards opposite focal states.

Trajectory analysis

For example, if the focal states of two adjacent *regular* domains D_1 and D_2 (see figure 2.9c) makes the trajectories point towards different directions in one of the dimensions, this may result in a *gliding* behaviour along a threshold plane or *switching* domain. Where threshold planes themselves intersect (e.g. forming a line), this can result in (numerically complicated) lower-dimensional gliding, until a steady state is reached, or until a next focal state leads the system to higher dimensions again. As with ODEs, analysis of these trajectories again leads to the discovery of the system's steady states and limit cycles.

2.2.7 Qualitative piecewise-linear differential equations

A model in the PLDE framework can also be studied purely qualitatively, by translating it into a Qualitative Piecewise Linear model, and by running a qualitative simulation and analysis (de Jong 2004). To this end, each of the phase space (hyper)boxes (see section 2.2.6), and each separating threshold plane, as well as each separating edge fragment and point, is associated with a qualitative state, as in figure 2.9a. The purpose of qualitative simulation is then to generate, and subsequently analyse, a state transition graph for the gene regulatory network. This graph contains and connects all logical states that can possibly be reached, starting from one or more given initial logical states.

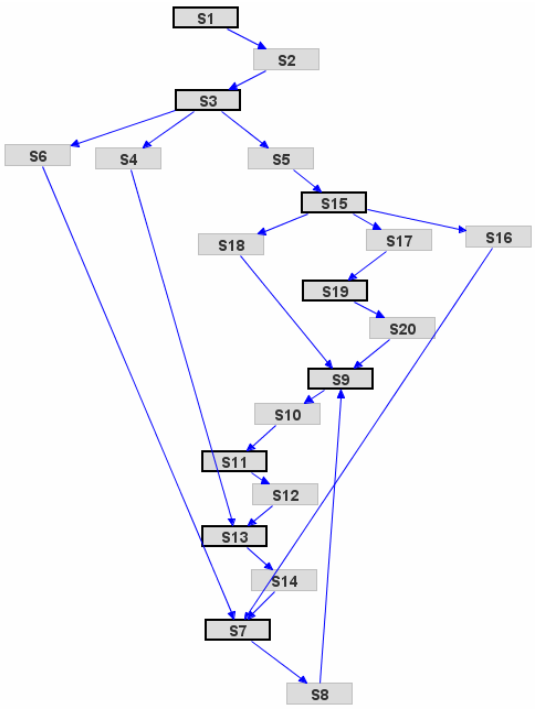


Figure 2.10. Qualitative state transition graph. This state transition graph is generated with the GNA software for our running example model defined in figure 2.7b, and starting from 1 of its 16 possible initial states. Boxes representing a regular domain have labels with bold borders; other boxes are switching domains such as the 2-dimensional threshold planes. The layout of the boxes has be rearranged somewhat to improve the presentation of the logical flow. Note the implied cyclic behaviour formed by the regular states labelled S7-S9-S11-S13.

Example

For example, figure 2.10 shows a state transition graph generated by the Genetic Network Analyzer software (de Jong 2003) for the simple three-component network defined in figure 2.7b, starting from 1 of the 16 possible initial states, namely: " $\theta_{11} < x_1 < \theta_{12}$, $0 < x_2 < \theta_{21}$, $0 < x_3 < \theta_{31}$ " (this box is positioned two boxes left of the highlighted box in figure 2.8a). Given this initial state, the system will reach cyclic behaviour (regular states with generated labels S7-S9-S11-S13, separated by threshold domains), no matter what path was followed to get there. These four boxes are all domains with $x_1 > \theta_{13}$, and correspond to the spiralling behaviour mentioned in figure 2.8a.

Cautions
The main weakness of qualitative simulations is their lack of scalability: the state transition graph grows quickly out of bounds. This limits qualitative analysis to fairly small regulatory systems, far more modest than the dozens of components that can typically be handled by quantitative models. Also, next to defining the network structure by entering the equations of figure 2.7b, one still has to enter a number of focal states (here: κ_1/γ_1 , κ_2/γ_2 , κ_4/γ_2 , $(\kappa_2+\kappa_4)/\gamma_2$, κ_3/γ_3). This is the qualitative counterpart of defining rates and threshold values in the quantitative framework, but arguably less intuitive. Since defining the focal states is a task more oriented towards mathematical considerations, it tends to put some limits on the immediate applicability by experimental biologists.

2.2.8 Spatially distributed models

The models described above typically have limited application in multicellular surroundings. They assume that regulatory systems work in a spatially homogeneous environment. While this assumption may be allowed in some cases, often one must take into account that cells consist of different compartments and that biomolecules are diffusing or are being actively transported between the separated regions. Or, when studying systems at a larger scale, one must take into account that there are multiple cells that all have a different regulatory state and that can influence or communicate with each other. In the latter case, this can result in interesting pattern formation which is crucial in e.g. embryonic development and morphogenesis. To model this spatial heterogeneity, one can run multiple simulations in parallel and define through which components they interact. Alternatively, one can also apply an extended version of the aforementioned original ODEs, and model both biochemical reactions and spatial diffusion in a uniform set of equations. We begin by describing the latter method.

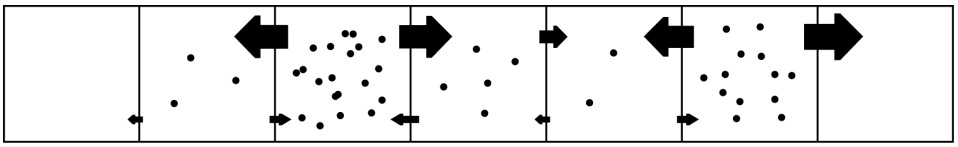


Figure 2.11. A one-dimensional, multicellular, spatial configuration. Illustrated is the diffusion proportional to the concentration, of one the components (gene product, hormone, etc) to and from neighbouring compartments.

Reaction-diffusion equations
For a start, one may assume that the spatial configuration is fixed and that the diffusion of gene products occurs proportionally to their concentration differences with neighbouring compartments; like in the one-dimensional example of figure 2.11. This is handled mathematically by taking the original full ODE equations for one cell, and adding both terms for the efflux and terms for the influx, to and from neighbouring compartments. In a simple system like figure 2.11, these *reaction-diffusion* equations can already be applied to generate emerging patterns. For example a locally self-activating gene product that laterally inhibits another gene can cause component distribution polarization, or a pattern of separated spikes (Meinhardt 1974, 1982).

But one may also consider a much larger number of compartments. One may even look at them as infinitesimally small. Then the differential equations' terms for influx and efflux can be seen as spatial gradients and are replaced by partial derivatives in the spatial dimension(s). This makes the equations *partial differential equations*. Having one set of differential equations for the entire occupied space allows the efficient study of activator-inhibitor systems. For example, in the early embryonic development of the fruit fly Drosophila, the presence of particular gene products in particular locations suppresses the presence of specific other gene products. This leads to multiple gradient distributions of the different gene products. Eventually, this controlled molecular diffusion and localization determines at which places the different body parts of the fly will develop.
For more in-depth discussions of reaction-diffusion equations and biological pattern formation, and some application examples, see (Gierer 1972, Holloway 2007, chapter 5 in Meinhardt 1982, Schvartsman 2002).
Note that the assumption of proportional diffusion makes the equations in their simplest form not directly applicable to actively transported or pumped components. For example the plant hormone auxin's transport direction is under control of a sophisticated molecular steering machinery.

Spatial models
Next to understanding the gene regulatory workings of cell growth and differentiation, it is also interesting to dynamically visualise their large-scale developmental effects, in a growing spatial model. For this, physical simulation frameworks exist and are being developed to model the spatial growth and organization of cells, and the formation of organs. Simple rules to drive cell expansion and proliferation, and their coupling with some molecular cues, can already yield remarkable results. But ultimately, these spatial models may be coupled with the full molecular basis of genetic network dynamics in each constituent cell, given sufficient knowledge and computer power. Multiple copies of the same set of differential equations would simulate the behaviour inside each cell, deal with the component diffusion or transport, and couple this with physical effects. However, as it is still a challenge to model a single cell it may still take some time before such a detailed 'virtual organ' becomes reality.

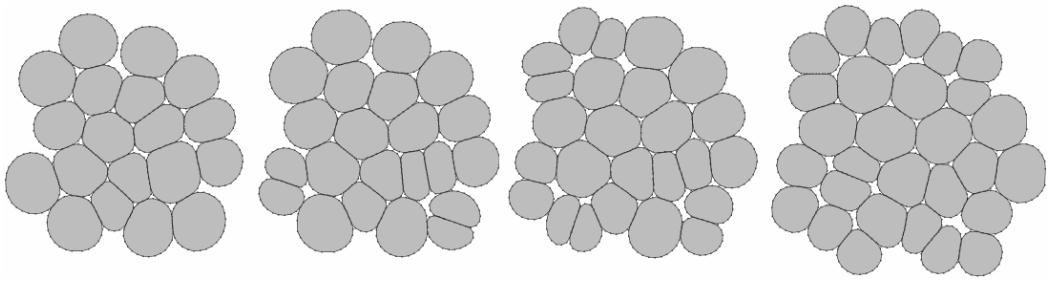


Figure 2.12. Four snapshots of a cell growth and division simulation. The simulation was based on predefined rules for cell volume expansion, cell membrane/wall extension, cell repulsion in case of imminent overlap, and cell division when reaching a critical volume. Cell boundaries were modelled by finite elements. The software used here to simulate cell proliferation, was programmed during a short exploratory excursion in our research project.

Spatial modelling of cell and cell membrane/wall growth and division is often first simplified to two dimensions to study general principles of pattern formation (see figure 2.12), or it is applied to near-2D systems like leaf surface growth and cell differentiation

into veins, stomata, etc. (Merks 2007). More complex studies use all three spatial dimensions; or they start with hybrid forms, e.g. surface development in 3D (Holloway 2007, Kaandorp 2001, Merks 2003, Merks 2004).
One of the various techniques for spatial model simulation is the straightforward numerical integration of mechanical physics: cell dynamics are based on the previous time-step, while forces are coming from cell border compression and stretching, and growth pressure comes from cell volume parameter increase (see figure 2.12). Another technique is global energy optimization. This is a non-deterministic search algorithm for the (time-dependent) energy optimum, where cells and cell walls try to reach an as uncompressed and unstretched state as possible, as used in (Merks 2007). As always, special considerations must be made to prevent spatial 'invasion' of outgrowing cell borders into previously unconnected structures.
Let it be noted that these spatial models to simulate growth tend to have difficulties with long-range interactions that would involve cell-block shifting (e.g. the leaf tip in its entirety should be pushed significantly, as long as the leaf base keeps growing). Also, the limited quantitative knowledge connecting the molecular and the cellular scales, as well as the computational tractability still pose challenges.

2.2.9 Stochastic modelling

The stochastic nature of gene regulation
When we look closer at the biological process of gene activation, we see that even the detailed ODE differential equations are only an approximation. First, they assume that gene activation happens in a continuous manner. And second, they assume that the process is deterministic. Both assumptions are in fact not true. It takes a random process for a gene transcription activator to locate and eventually bind to a gene's promoter; and these temporary binding states only last during discrete time intervals, until they break up again (Gibson 2001, Zhu 2007). In the mathematical formalism of *stochastic* modelling, the continuous variables that describe the components in ODE equations are now replaced by discrete amounts of molecules. Also, a probability function is now applied, which expresses the probability that a system in a certain configuration (with given discrete amounts for each component) will evolve into different other configurations over time. This leads to a so-called *stochastic master equation* (de Jong 2002, van Kampen 1997).

Stochastic modelling versus ODEs
The choice for a stochastic model usually depends on the abundance of the studied components. When a transcription factor's abundance is very low, in the order of only a few copies per cell, then the effect of the discrete and indeterministic nature will emerge, justifying the choice of using a stochastic model. But when one observes a large number of molecules, then the discreteness and randomness will usually level out, and the approximation made by a differential equation model will usually work fine. There is, however a noted exception when key decision points are determined stochastically, e.g. in the case of λ-phage described below, where the difference with ODE remains pronounced.

Simulation and applications
One can also simulate the time evolution of a stochastic system (e.g. Adalsteinsson 2004, Salis 2006, Wu 2007). With a large number of simulations one can examine the probability distribution of various outcomes. For example, (McAdams and Arkin 1997) have carried out stochastic simulations to study how the stochastic nature of one gene's regulation influences the timing and frequency of its expression. They discovered that this occurs in short bursts of gene product output, and at random time intervals. They described how such random gene expression can produce a probabilistic outcome in a switching mechanism that selects between competing pathway regulators; for example, for the bacteriophage λ's decision to switch between its lysogenic and its lytic pathway.

In conclusion, even though stochastic modelling comes closer to the molecular reality of gene regulation, it requires detailed knowledge of the regulatory mechanism (for the probability function). Whether the benefits cover the extra costs (the computationally intensive simulations and the detailed molecular descriptions), depends on the level of detail one wishes to acquire about the regulatory process. On a large time-scale and with large gene product abundances, stochastic effects tend to level out, and differential equation models may give a sufficiently good answer.

2.2.10 Petri-nets

Originally, Petri nets were designed to model various man-machine interaction systems, like processes of manufacturing and communication (Petri 1962). But due to their graph-structure, they have also become a convenient mathematical formalism that allows an intuitive representation of biochemical networks (Chaouiya 2007, Simão 2005). A Petri net is a directed graph that contains two kinds of nodes, and its edges only run from one type of node to the other type. See figure 2.13; the two node types are the circles and the filled bars here. The first type of nodes is called *places* and represents biological entities (like proteins and metabolites). The second type is called *transitions* and usually stands for biochemical reactions. In addition, each place-node can be filled with a number of *tokens* that represent the discrete abundance of that place's biological entity.

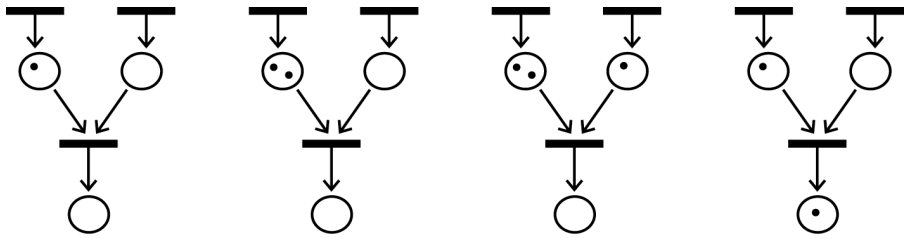


Figure 2.13. Petri-net simulation. Four consecutive steps in a mock-up Petri-net simulation. The top two transition-nodes can put one token at a time in their connected place-node. The lower transition-node is programmed to take a token from each of its input places, while inserting one token in its output place.

Dynamics simulation
Simulation of a Petri-net happens by letting the transition-nodes (the reactions) consume some (or none) of the tokens in its connected input place-nodes and produce tokens into its output place-nodes. Like that, this process leans close to the biological reality of

discrete gene product creation in the presence of transcription factors and inhibitors. Also, in order to let the transitions happen asynchronously, the simulation is carried out non-deterministically, meaning that at each step, the next transition is randomly chosen out of the list of possible transitions at that point. Note that this discrete and random process is reminiscent of a stochastic simulation.

Petri net extensions
Next to this basic representation, Petri nets also allow different levels of abstraction. An entity (place) may represent a gene, RNA, or protein, but also just an 'activated gene', or even an entire 'active biochemical process'. Also, various technical extensions to the original formalism have been implemented. Hybrid Petri nets include *continuous* entities and continuous transitions, so as to establish a link between qualitative and quantitative aspects (Matsuno 2000, Nagasaki 2004, Troncale 2006), and Stochastic Petri nets implement randomized time delays before transitions occur (Goss 1998). This stochastic application is a natural one since one can use a one-to-one mapping of the tokens and places to discrete molecular abundances (Srivastava 2001).

Petri Nets offer the advantage of connecting different scales of abstraction, generating both qualitative and quantitative results, and more depending on the extensions used. But while they generate these simulation results, it should be noted that their systemic analysis poses more difficulties than e.g. the ODE formalisms.

2.2.11 Rule-based formalisms

The rule-base formalism differs substantially from all the modelling techniques above, since it does not work with variables that represent component amounts. Instead, rule-based systems work with a large set of rules that can connect entities of disparate kinds and of disparate levels of scale; and in addition, rule-based simulations are event-based. In fact, evaluating a model in this formalism leans more towards artificial reasoning than towards simulation in the sense used until now.
For example, suppose we have a set of rules like "transcription factor A binds to promoter X", "gene B has a promoter X", "activation of gene B produces protein C", and "protein C causes cell division", plus a number of householding rules like "a gene with a transcription factor bound to its promoter becomes active". Then if the system receives the 'event' or fact that protein A is present in the cell, events will cascade through the model, and the system will decide that the cell will divide.
From this simple example, one can already see how different scales of information and different types of entities (protein vs. cell division) can all be described in the same formalism. For example, the HyBrow system (Racunas 2004, Racunas 2006) is experimental software that follows this formalism at the level of designing hypotheses and validating them against information (sequences of rules) already in the system. However, it is not obvious to define a complete system and think about every biological aspect that must be included as a rule. Also, it proves difficult to incorporate continuous quantitative data.

2.2.12 Conclusion

Whichever formalism you choose for modelling and simulation, it will always be an approximation of biological reality. The key for choosing a formalism is to keep in mind which level of detail or abstraction is appropriate for the system you are studying. If you want most detailed predictions and statistics about the subtle activation scenarios of one or two genes, you are likely to choose a stochastic model. If you want a rather detailed model for a process of tens of genes, you will rather choose a differential equation model. If there is plenty of quantitative data available, or if you want to perform a systemic analysis on these genes, ODEs may seem appropriate. But if there is limited quantitative data (which is usually the case), like only a number of gene expression profiles under a few different circumstances, a simplification of the ODEs will be appropriate, like PLDE simulation or qualitative analysis. You can even go to more coarse levels of abstraction and deal with larger numbers of genes, and analyze their expression with Bayesian networks, in a hunt for general or conspicuous group-to-group interactions. Or you might want to mix levels of abstraction, with Petri nets or rule-based models. Considerations like these make it clear why there are so many modelling techniques, and that each method has a specific window of applicability. Many of these frameworks are not so accessible to biologists, hence the initiative described in chapter 3 was launched.

Chapter 3

SIM-plex: Genetic network simulator

3.1 Rationale & Core of SIM-plex functionality

One of the aims of our research has been to build bridges between the wet-lab experimental results, and dry-lab systematic analyses. To this end, we built and applied a number of software tools to collect pieces of information and integrate them in a larger biological overview. SIM-plex is one of these tools. To describe SIM-plex, we must discuss the two important aspects that made it fit for application in our various current research projects at the PSB department. These two aspects are: the choice of the mathematical formalism, and the design of the user interface.

3.1.1 Choice of the PLDE formalism

First of all, SIM-plex is a genetic network simulator based on the Piecewise-Linear Differential Equations (PLDE) formalism (de Jong 2003, de Jong 2004) (See chapter 2, section 2.2.6 for its mathematical details). As mentioned in chapter 2, a central question before building a model and running simulations, is to choose the appropriate modelling formalism for the biological problem at hand. In our case, we were dealing with fragmentary knowledge of the plant cell cycle (Inzé 2005, Inzé 2006, Stals 2001). We also had some examples of cell cycle network models from other species (Novak 2001, Novak 2004), but compared to them, the plant cell cycle knowledge was still less advanced, more incomplete. Therefore our first goal was to study the general behaviour of the regulatory network, based on the genetic players that were known, while being able to make educated guesses about gaps in the network. Also, only a limited amount of quantitative information was available. Almost no biochemical reaction rates were known. But still, experimental techniques like microarrays and Western blots gave us some information about mRNA expression and protein activity profiles over time. In this respect, the formalism of PLDEs turned out to be an excellent choice, as it allows modest quantitative modelling that can be connected with a qualitative interpretation (Note that the purely qualitative sibling of the PLDE formalism of section 2.2.7, was not preferred because of scalability issues discussed earlier). Furthermore, the logical, switch-like nature of gene activation in the PLDE model left the door open for building an 'if-then' representation

layer on top of it, that biologists could easily connect with (see below). This made modelling accessible for biologists and stimulated the mutual exchange of ideas.

3.1.2 Design of the User Interface

The SIM-plex user-interface forms a bridge between the wet-lab biologist and the mathematical world. The rather incomplete knowledge of many regulatory systems that are to be modelled, urges for a close and sustainable involvement of biologists most deeply familiar with the latest biological knowledge in the field. Therefore, in order to enable an intuitive design or definition of a network, it was necessary to build a certain shield around the pure mathematical formulation of differential equations, some layer of natural abstraction on top of the mathematics. It turns out that the mathematics of PLDEs allows just such an extension. The switch-like nature of how PLDEs approximate gene activation (see also figure 2.6b), can be directly translated into the logical way of how people usually reason about gene activations. This switch-like on/off behaviour lies in the Hill function that is used in all PLDEs. This function allows defining that when the abundance or activity of one gene accumulates above a critical threshold, a second gene's transcription gets switched on, or that it at least gets a step-wise increase or decrease in its activation (see section 3.2 for details). This type of reasoning readily translates to and from 'if-then' statements, like: "*if* gene A is active enough, *then* gene B becomes active too". Or, since PLDEs are still a quantitative formalism: "if gene-product A rises above a threshold *x*, then gene B is produced at *y* more units per time-unit" (*y* can be positive or negative).

So in essence, all a person should now do to define a model of a regulatory network is to provide a list of such if-then statements. The user interface of SIM-plex accepts this list and derives, behind the scenes, a set of PLDE differential equations from it. Then SIM-plex runs a simulation for the equations and shows the results as a set of gene-product profile plots.

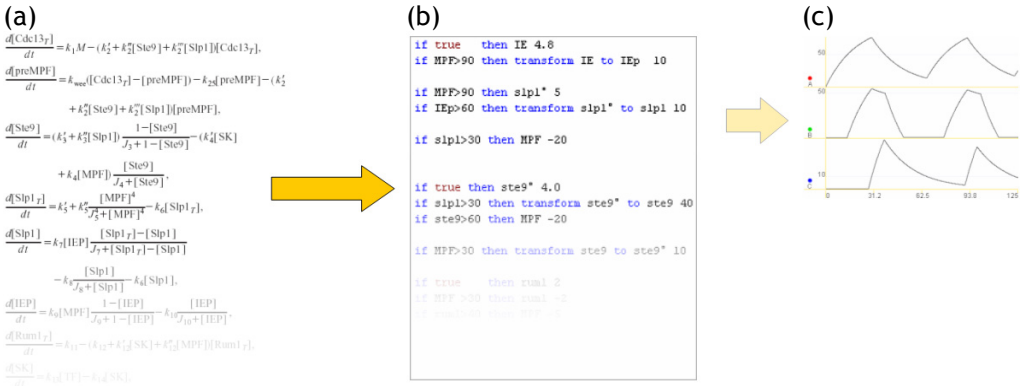


Figure 3.1. ODE equations compared to the logical SIM-plex statements. (a) Part of a list of ODEs from a yeast cell cycle model (Novak 2001). (b) For the same model, a list of logical statements as how they are given in SIM-plex, intuitively much better comprehensible. SIM-plex prevents having to formulate the ODE equations and only requires the logical if-then statements. (c) Plot of the simulation results generated by SIM-plex.

Figure 3.1 illustrates how SIM-plex makes modelling and simulation more accessible for scientists with a less mathematical background. It show a comparison between a set of

differential equations used by Novak and Tyson to model the fission yeast cell cycle, and the set of logical if-then statements describing the very same model, as entered in SIM-plex. It demonstrates that SIM-plex takes away the burden of the purely mathematical formulation. This allows biologists to better focus on the logic of the model, and also to tweak it by easily adding, changing or removing hypothetical interactions.

This approach of SIM-plex to connect the biological logic and a mathematical formulation has been successful, and created clear synergy. It helped to form an improved understanding of both the biological and the systematic (mathematical, informatical) aspects of a number of systems biological studies.

We gave our software the name "SIM-plex", which stands for:
- SIMulating genetic networks (→ to model and simulate gene regulatory networks)
- with Piecewise Linear Equations (→ the basic mathematical model)
- in Comfortable Statements. (→ refers to the user-friendly interface)

Summary
With SIM-plex, we have taken existing mathematical techniques and dressed them up as a system usable by a biologist. We have built a tool to bring the mathematics closer to molecular biologists, since they are best placed to reason about biomolecular networks. Like this, we brought the spirit of Systems Biology closer to the workbench. Our approach has shown its success in the lab, as biologists have been able to place together components to model a number of processes including the plant endocycle network (unpublished and published results of Beemster, De Veylder). In this respect, the development of SIM-plex was also a study demonstrating how to create usability when introducing a mathematical model analysis tool in the experimental biologist's world.

3.2 Core SIM-plex functionality

3.2.1 Core PLDE / 'if-then' functionality

We describe here the mathematics behind the connection between PLDEs and the logical 'if-then' statements. This was described in (Vercruysse 2005), but only briefly due to the journal's article length limitation. Below, we provide full mathematical background.

Basic gene activation
For a start, just consider one gene activating another gene. Then for the activated gene the PLDE equation (see section 2.2.6) :

$$dx_i/dt = \Sigma_{j\in L}\ \kappa_k s_k(x_j, \theta_{jm}) - \gamma_i x_i$$

without degradation rate ($\gamma_i = 0$), and with only one positive Hill equation:

$$s^+(x_j, \theta_{jm}) = 0 \text{ for } x_j < \theta_{jm}, \quad \text{but 1 for } x_j > \theta_{jm}$$

reduces to

$$dx_2/dt = \kappa . s^+(x_1, \theta)$$

or alternatively

$$dx_2/dt = 0 \text{ for } x_1 < \theta, \text{ and}$$
$$\kappa \text{ for } x_1 > \theta.$$

which reads out as:

If gene product 1 is present above a *threshold* θ,
then gene product 2 will be produced at a *rate* κ.

or freely translated:

If gene 1 is active enough, then gene 2 becomes active too.

In a SIM-plex statement, this could look like:

if A>20 **then** B 5

where 20 is the threshold for A's activation, and 5 is B's production rate when A is active. Units for these values can be concentrations or amounts, and are user-determined. Parameters like these can be established based on experimental information. Or since this information is in most cases not available, educated guesses can be made so as to match observed quantitative time-profiles or a qualitative behaviour (micro-arrays, protein abundance). This procedure has also been used by Novak and colleagues (Novak 2001, Novak 2004).

A gene's promoter, the regulatory region that precedes it in the DNA, usually contains not just one motif, but a complex series of them. These motifs are regions where other regulatory molecules, such as proteins, can bind in order to enable, augment, attenuate or block the associated gene's transcription. SIM-plex allows modelling this via two methods: combined conditions, and statement additivity.

Multi-condition gene activation

When for example a gene is only activated when components A and B are active, but C is fairly inactive, then a combined condition could look like:

if A>10 **and** B>20 **and** C<2 **then** D 5

SIM-plex translates this combined condition to a product of step-up and step-down functions, the positive and negative Hill functions s^+ and s^-. The mathematical translation will look like:

$$dD/dt = 5 \,.\, s^+(A,10) \,.\, s^+(B,20) \,.\, s^-(C,2)$$

where the right-hand side only yields 5 under the conditions given in the if-then statement.

Statement additivity: enhancement

Sometimes when a gene is activated, its activation can still be enhanced, its transcription rate increased, when an extra component binds to a specific promoter motif. This stepwise increase in activation can be modelled in SIM-plex thanks to *statement additivity*. This means that when two statements evolve to *true* for the same target gene, both creation rates are added together. For example, consider the statements

if A>10 **then** C 5
if A>10 **and** B>10 **then** C 2 .

Then if A rises above 10 units, C's gene-product gets created at a rate of 5 units/time-unit. But then when B also rises above 10 units, both conditions are fulfilled at the same time. Then C gets an extra creation boost of 2, resulting in a creation of 7 units per time-unit.

Basic gene deactivation

Gene deactivation is modelled by using a negative creation rate, for instance:

if A>10 **then** B -5

Note that if a concentration value would go below zero at any timestep in the simulation, SIM-plex will set it to zero. Also note that for deactivation, no step-down Hill-function s^- is used, as this would give the undesirable mathematical side-effect of an extra activation when the deactivator is below the deactivation threshold.

Statement additivity: attenuation

Gene attenuation, or partial interference with activation, is established via the same method of additivity as demonstrated in the enhancement case. For instance:

if A>10 **then** C 5
if B>10 **then** C -2

will create C at a rate of 5 units/time-unit when A is active enough, but at a rate of 3 if A and B are both active. (As an aside: note that if only B is active, C will not drop below zero, but stay at zero. As mentioned before, this is because SIM-plex doesn't allow negative concentrations. In order to prevent this artificial dependence on a cut-off to 0, one can simply use a multi-condition like "if A>10 and B>10" here in the second statement.)

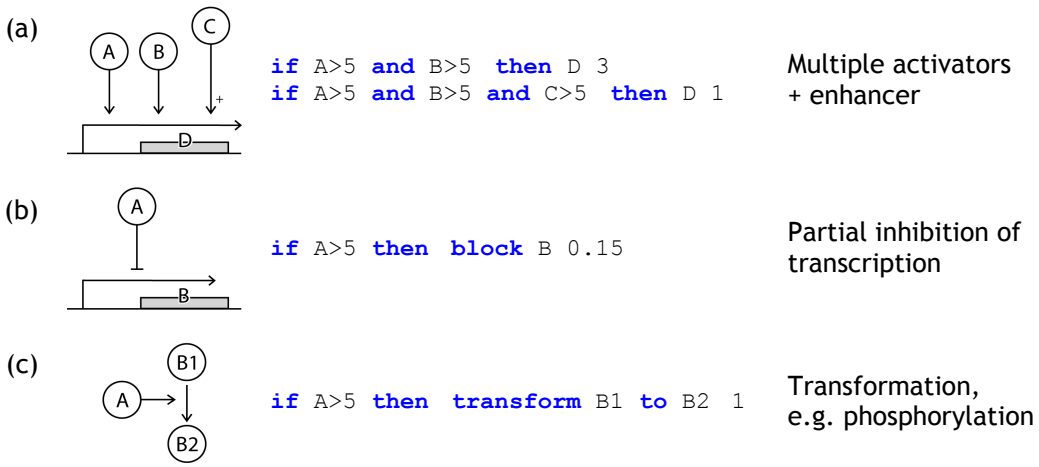


Figure 3.2. Examples of statements used to define genetic networks in SIM-plex. Here we show a few commonly used graphical representations of genetic network model interactions, and their translation into SIM-plex statements. (a) Component creation and enhancement, (b) block and (c) transformation are defined as described in the main text of sections 3.2.1 and 3.2.2.

3.2.2 Additional basic SIM-plex statements

Activation block

It should also be possible to completely block a gene's activation altogether. In some cases a gene's creation can at any time be regulated by a variety of different combinable influences. Then when a strong inhibitory component shows up, it must be able to block all creation influences that are working at that time. For this, one could just give it a creation rate of say -1000. But more cleanly (and for a reason mentioned a few sections later), SIM-plex allows to use the keyword "block". For example in:

if A>10 **then block** B

will completely block B's creation, no matter what its other transcriptionally regulating influences are. Note that this would not make B disappear instantaneously, as only its creation is blocked. Alternatively,

if A>10 **then block** C 0.20

will partially block creation of component C, reducing it to 20% of its unblocked creation rate. A few examples of statements described until now are given in figure 3.2.

Everlasting truths

Sometimes when one runs a genetic network simulation, one is obliged to assume that a certain gene product is always being produced somehow, without being able to give a known cause. For this, it should be possible in SIM-plex to use a statement with a condition that always evolves to *true*. This is done by giving the keyword "true" as the condition after the "if" keyword, as in:

if true **then** A 5 .

As a result, A will be produced at a rate of 5 throughout the simulation. It may of course still be attenuated or blocked by other, additive influences.

Note: one may suggest that this is a rather mathematical construct, a bit further away from the biologist user, and that one should rather provide a statement like "always A 5". On the other hand, it is also true that a lower number of core statement types makes definition of models less cluttered and easier to use. In the spirit of this, SIM-plex doesn't use the "always" keyword, but rather the "if true then" construct, so as to keep everything in the if-then statement format.

Transformation

First a word about RNA and proteins. The answer to the question as to what the *gene products* in a model precisely represent, mRNA or proteins, is sometimes arbitrary. One often assumes that protein concentrations are reflected by mRNA abundances. This assumption is for instance frequently made, with caution, during the interpretation of microarray results. Understandably, as global measurements of protein amounts and their activity are currently still much harder to carry out.

So when using any modelling tool, one often can, again with caution, model a 'gene product' as if it has both mRNA and protein qualities. This means that this gene product can be created under the regulatory influence of other gene products (usually proteins), as described above, but at the same time that it can undergo protein biochemical reactions, like phosphorylation or binding into a protein complex.

Another consideration pertains to the approximation of gene transcription by step-functions, which was shown to be valid in (Glass 1973). One could extend this assumption in genetic networks, by also including other biochemical reactions like protein transformations (like phosphorylation). One could approximate that the start of a protein's modification also happens under a switch-like influence. This just brings the model one step closer to a generalized logical model (as described in section 2.2.4). SIM-plex helps defining such protein modifications by introducing the "transform" keyword. It is best illustrated with a few examples. For instance, the phosphorylation of a gene-product B (here in the protein sense), can be defined in SIM-plex as:

if A>10 **then transform** B **to** Bp 5

where B is the unphosphorylated form of the protein, and Bp is the phosphorylated form. Note that Bp is a new biochemical molecule derived from B by addition of a phosphate (PO_4) group on some amino acid of the protein chain. An if-then-transform statement is internally translated by SIM-plex into two basic if-then statements: one to model the source component's disappearance, and one to model the target components creation, at

the same but opposite rate. In the example, SIM-plex will internally replace the statement by the following two statements:

if A>10 **then** B -5
if A>10 **then** Bp 5 .

In a similar manner one can also model protein complex formation. This statement:

if X>10 **then transform** (A A B) **to** Complex 5

is internally translated in SIM-plex to these equivalent statements:

if X>10 **then** A -5
if X>10 **then** A -5
if X>10 **then** B -5
if X>10 **then** Complex 5

which show how one unit of Complex is created when at the same time two units of A and one unit of B are removed.

Component declaration

Before a component name can be used in the if-then definitions, it must be declared, or identified, first. This is done via a simple "comp" statement. Also, as the SIM-plex simulator engine needs to know what the initial abundance of the component is, it can be defined immediately in this statement. For example:

comp A 10
comp B

declares that the system has two components, one named *A* and having an initial abundance of 10 units, and the other named *B* and having an initial abundance of 0, which is the default value when the number is omitted.

Furthermore, this statement can be used to define a non-standard degradation rate for the declared component. The default degradation is set to a small value of 0.05, meaning that 5% of the component is degraded or destructed during each time-unit (cf. section 2.2.6). But a statement like for instance:

comp C 100 0.20

could be used to declare a component B with an initial presence of 100 units, that undergoes a strong degradation influence of 20% each time-unit. Using such a degradation mechanism is justified when a component's disappearance happens proportional to its abundance.

In summary, the "comp" statements are needed to declare each component name used in the model, and can be used to define non-zero initial abundances, and non-default degradation rates.

3.2.3 The simulation engine

Once a model is defined via a list of if-then statements, SIM-plex is able to combine them all and build a set of piecewise-linear differential equations based on them. With these, SIM-plex can run a simulation to determine the dynamical behaviour of the regulatory network. This result is then shown to the user as plots of gene product amounts over time.

Numerical integration

For the simulation of the PLDEs, SIM-plex uses the Euler method. The Euler method is a basic method for numerical integration of differential equations. It starts from a given set of component amounts, the *initial state*, and goes forward through time in small, equal time-steps. At each considered time point the differential equations are evaluated, for instance "$dx_i(t)/dt$ = C". This derivate of $x_i(t)$ indicates the local change in direction and size for that component. The actual change for each component dx_i is then "$C.\Delta t$", with Δt the time-step size.
SIM-plex uses a default small time-step for the integration, but one can also define a custom one, if one needs finer or coarser granularity. This is done via the "timepoints" statement, for instance:

timepoints 0 **to** 200 **step** 0.01

Note that a "timepoints" statement should always be included in any if-then statement list (but not necessarily a "step"-tail), because the simulator always needs to know over what time interval it should perform the integration. Such a basic statement could be:

timepoints 0 **to** 100

To complete the numerical integration's description, it must be mentioned, as before, that at each time point, the component amounts that would be negative are reset to zero. This could be seen as a strong, implicit if-then condition that creates the component as soon as it drops below a threshold of zero.

SIM-plex's solution to threshold hyperspaces
A basic PLDE formalism does not define the value of the equations on the threshold planes or hyperspaces. Given the mathematical quandary of trying to integrate the PLDE equations along these regions using mathematical exactness, and given the fact that this formalism is an approximation of the rather stochastic biological reality anyway, we decided to make a small adjustment to the PLDEs to avoid this unnecessary trouble (see section 2.2.6).
SIM-plex defines the phase space compartments (see figure 2.8a and 2.9a) in such a manner that they include their lower threshold: $\theta_{ij} \leq x_i < \theta_{ij+1}$. As a result, the original complex gliding behaviour along a threshold will then be straightforwardly solved by quantitative numerical simulation via jumps back and forward over that threshold. The effect on the outcome of the simulation will be minimal, as the exactness of PLDE differential equations is only an approximation of the more stochastic nature of biological reality anyway (see section 2.2.9 about stochastic modelling).

3.2.4 The user interface

The graphical user interface, as shown in figure 3.3 and figure 3.x, provides the user with one window to enter a list of statements and declarations (left), and one or two windows that show the plots resulting from a simulation run (right). In figure 3.2, only the Single Plots window is shown, while the Combined Plot window is hidden.
As an example, we show the SIM-plex statement definition of the running model used in section 2.2.6, figure 2.7, including the plots of a simulation run for this model. In practice, the modeller types in a list of if-then statements (plus the comp declarations and a timepoints range), and presses F5 (or uses the menu) to run a simulation and see the results.

```
comp x1 15
comp x2
comp x3

if x1>10 then x1  5
if x1>20 then x2  5
if x2>10 then x3  5
if x1>30 and x3>10 then x2 -8

timepoints 0 to 50
```

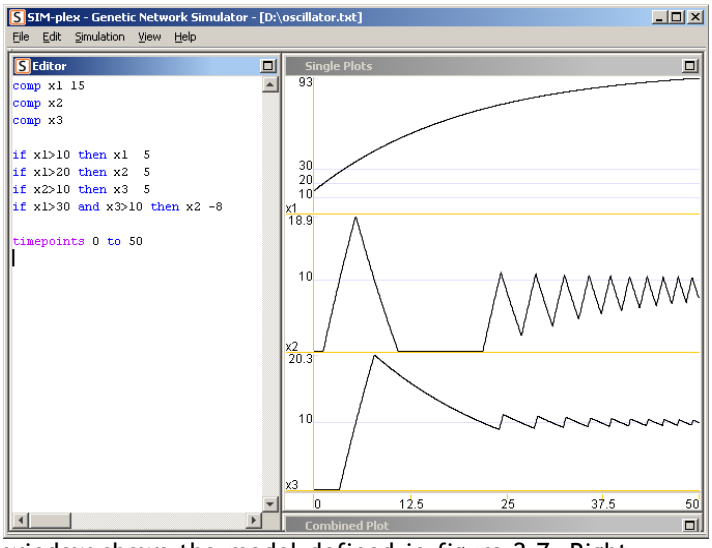


Figure 3.3. SIM-plex use case. Left window shows the model defined in figure 2.7. Right window shows a plot of the simulation results.

Syntax highlighting

As shown in figure 3.3, the statements are decorated with syntax-highlighting (as is the case with many programming language editors, and enabled by the public plugin 'jEdit 2.2.1 syntax highlighting package'). While numbers and component names and are left unchanged, SIM-plex keywords are shown in a conspicuous colour, different for every other class of keywords. This makes the statement definition much more visually clear and readable, and allows a quick detection and correction of typographical errors.

Error detection and reporting

It is possible that the modeller made a syntactical error in the statement list. In that case, when he/she tries to run a simulation, SIM-plex will detect the error, will make the cursor jump to the location of the error in the model definition window, and will report some explanation about the error in the plot window.

For the other features provided by the user interface, we refer to section 3.7 (appendix B) in this chapter.

3.2.5 Various remarks

As a conclusion to the core SIM-plex functionality, we discuss a number of considerations, based on some commonly asked questions about SIM-plex.

The switch mechanism

Note the difference with the Boolean or the generalized logical formalisms. With logical functions, one defines for example that when A and B are active, then C becomes immediately *active at a level x*. But With PLDEs, and so also SIM-plex, one would define that when A and B are active, then the *production of* C is switched on or increases with a *rate* of *x* units per time-unit. And so here, as soon as C is switched on, its abundance starts to slowly (or quickly) increase.

Degradation curves
When looking at the plots in figure 3.2, some people ask why the plot of for example component *x1* is bent instead of straight. As should be clear by now, this is caused by a (definable) degradation rate. As it is apparent, this causes active components to reach a saturation level, a steady state of creation equal to degradation. This is what should be expected, as no biological component has yet been observed to rise in amount or concentration forever.

Systems Biology
There are good reasons for modelling and simulation of a biological system, as already mentioned in section 2.1. To rephrase the essence: simulation allows one to consider the dynamics of the system. By defining and evaluating the connections and dynamics of a larger number of components in a trial-and-error way, one learns how models behave in a systemic way, which stimulates the creation of hypotheses, of missing links etc. Also, one can check if the current knowledge is sufficient to explain modifications of the biological system, or if some of our model knowledge is still incomplete. For instance it is easy to perform an *in-silico* deletion or modification of a gene in the model, and see if the results still comply with the observations made on a mutant, or transgenic line (see examples in chapter 4).

Parameter estimation
One still has to estimate some parameters, but compared to full ODEs, there are fewer. One doesn't have to take into account the details of the interactions anymore, being the shape of the activation function as described in sections 2.2.5 and 2.2.6. That leaves one parameter less to estimate for each interaction in the set of differential equations. Instead, one can focus on more essential parameters for the logical wiring of the system: the activation threshold and the creation rate. Both can in some cases be important for concentration effects. In practice, these parameters are estimated so as to mimic or produce certain observed or intended semi-quantitative / semi-qualitative behaviour, like transcript or protein abundance time-series profiles.

3.2.6 Merit of SIM-plex

We have taken complex, existing simulation technologies and made them accessible to a broader range of modellers, including the ones who obviously know the most about the biological reality: the biologists themselves. This was established by providing them with an intuitive interface that is based on, and directly linked to a mathematical foundation of logical if-then behaviour. Moreover, this requires estimation of a minimal number of rate constants. Furthermore, the link-up of a quick feedback loop yielding simulation results enables them to intuitively play with the system and try out various possibilities. As a result, as chapter 4 will show, SIM-plex was able to prove and improve its applicability by and in cooperation with experimental biologists.

3.3 Example: construction of a small model

As an example of how a simulation tool like SIM-plex can easily and quickly generate results, we use figure 3.4 to describe the build-up of a small genetic network toy model. Each SIM-plex screenshot shows an editor with a network definition, next to the dynamic plots resulting from the model's simulation (both single plots and a combined plot). Each network definition starts with a declaration of the used components, an optional colour for them in the combined plot window, and an initial amount (omitted here, so zero by default). From (A) through (E), one interaction is added at a time, indicated with an arrow in the editor.

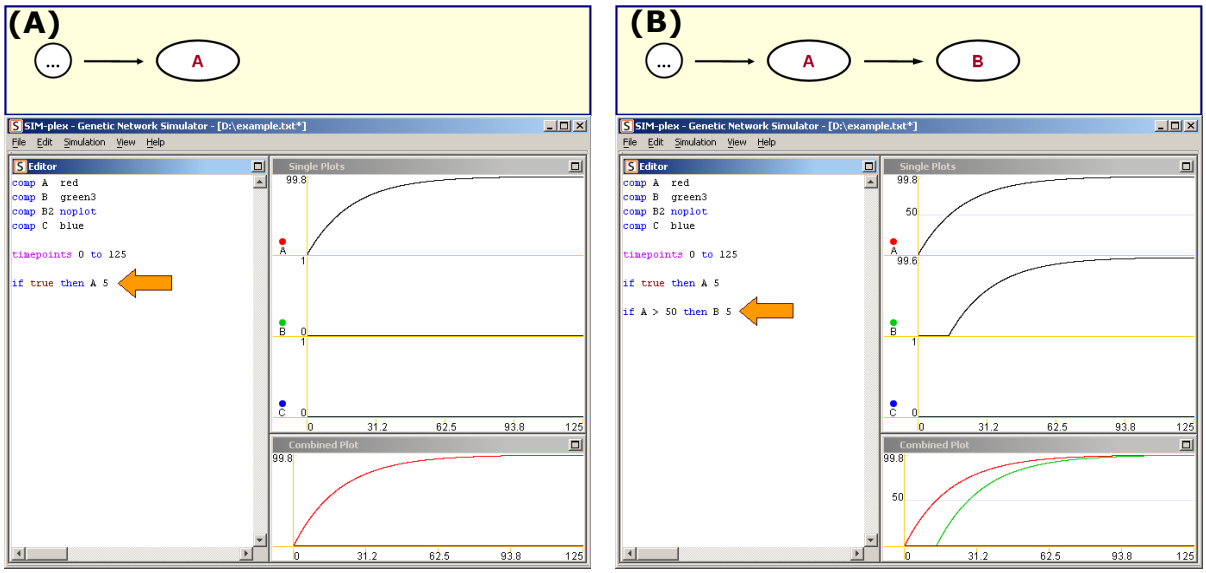
(A) As the first step we have a component *A* that is constantly activated by a not further explained cause (expressed as an if-true-then, within the overall if-then framework). The gene product A can represent RNA or protein, and its creation rate is 5 units per time-unit. *A* is also subject to a definable degradation percentage (this is 5% by default). *A* reaches saturation level when the degradation rate has grown as large as the creation speed. Here this is at a level of 100 units, when per time-unit, 5% is degraded = 5 units, and this precisely equals the 5 created units.

(B) Here we see component *A* activating *B*, or more precisely "if *A* is sufficiently active, then *B*'s creation is switched on. The activation threshold is defined here as 50 units. You can see that *B*'s gene product amount starts to rise as soon as its activator *A* has reached the horizontal line indicating its 50 unit threshold.

(C) This adds an extra interaction similar to the one above.

(D) This illustrates the interesting effect of introducing a negative feedback from *C* on *A*, through an "if-then-block" statement that models transcriptional inhibition. The system starts to oscillate. Note that without delaying intermediate step of (B) on *C*'s activation, the system would instead evolve towards equilibrium of *A* and *C*, just like *x2* and *x3* do in figure 3.3.

(E) This shows the definition and the peculiar effect of an extra phosphorylation of *B* under the influence of *C*.



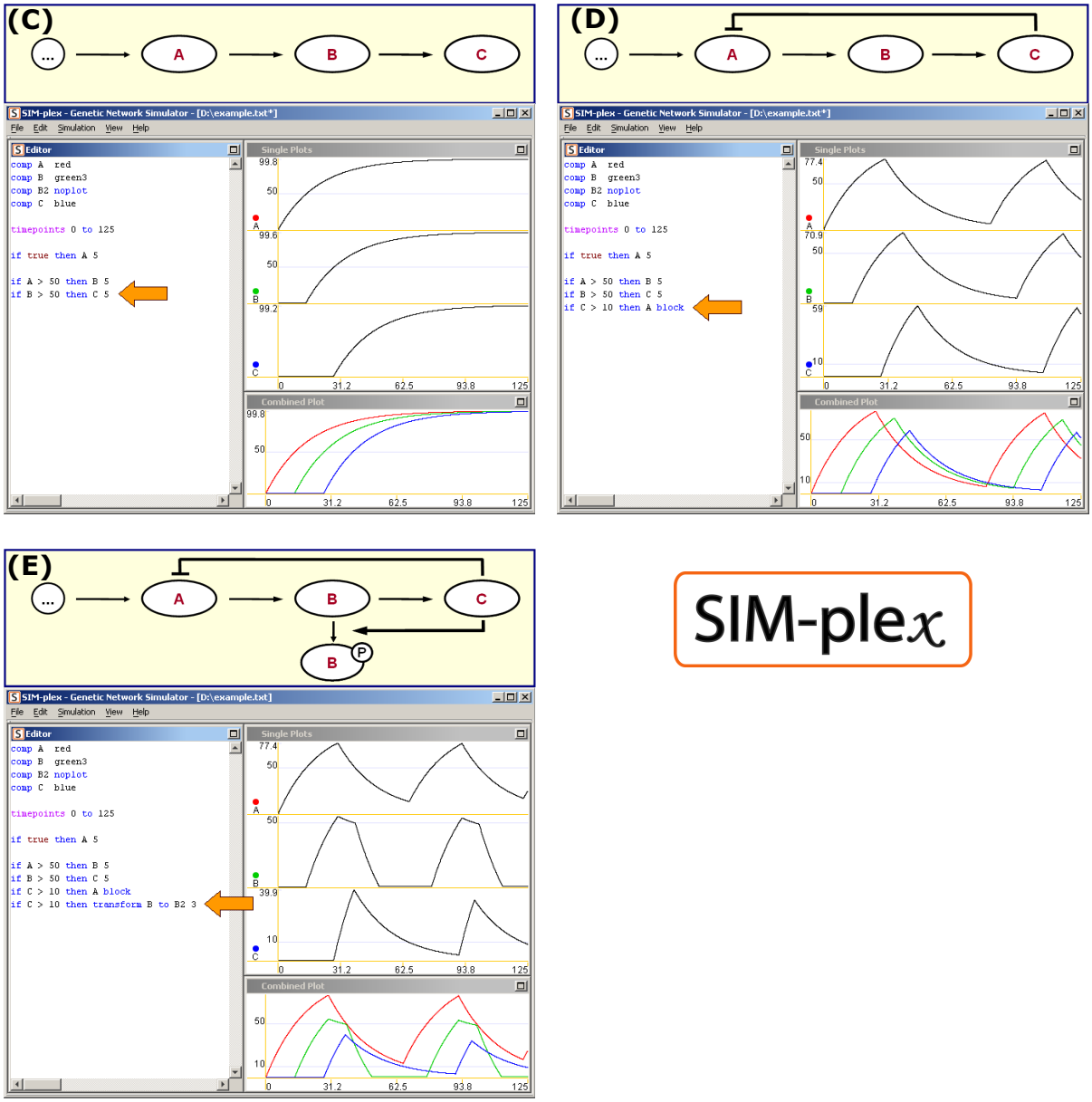


Figure 3.4. A sequence of simulations in SIM-plex. (A)-(E) The build-up of a small gene/protein regulatory network model. For every newly added interaction, a new simulation was run. Apparently, with a limited number of simple interactions, one can already define a fairly interesting network with cyclic dynamical behaviour.

While naturally much more complex models can be built, this example shows the basic concept, and an idea of how to build a model in SIM-plex.

3.4 Extra functionality

SIM-plex was applied to a number of topics investigated by molecular biologists at VIB-PSB (see chapter 4). Thanks to the direct interaction and subsequent real-life feedback, we could collect various suggestions and were able to lead SIM-plex to maturation. The most important, common, or specifically-needed feature-requests were programmed into the

software, even though some extensions go even a little beyond the basic PLDE formalism. In the following sections, we sum up the most important extensions.

3.4.1 Special components for models

Next to "normal" components that are defined via a "comp" statement and that represent a gene product (RNA or/and protein), it became apparent that some more advanced models need additional components. In the following sections we describe the need and use of three extra types of special components. These are *virtual components* (linear combinations of normal components), *fixed components* (having a fixed profile through time), and *mathematically defined components* (any mathematical function of other components and/or time).

Virtual components

The concept of a virtual component solves model-defining problems like the following: if a protein *A* is produced, and at the same time another molecule *B* is produced (for example, it's a less active derivative of *A*, like a phosphorylated version), but still both *A* and *B* have an additive effect on the regulation of *C*. How would one express this in statements like "if ...>... then ..."? One would need something like "if (...+...) > ... then ...". In order to avoid additional complexity in the if-then definitions, SIM-plex allows the definition of virtual components. In the example, a virtual component *V* could be defined as "A + 0.5 * B", and this virtual *V* would subsequently be the regulator of *C*.

In summary, a virtual component is a linear combination of two or more other components, and it can be used to model the additive effect of those components on a target component. Some example statements:

virtcomp Total_A = 0.9 * A1 + 0.5 * A2
if Total_A > 50 **then** C 5

Predefined (fixed) components

It occurs that a genetic network is under the driving influence of certain dynamical input. For example, experimental data may suggest that a gene's activity shows regular pulses; but one may not yet know what up-stream influences cause that behaviour. Yet that gene itself may be critical to steer a whole downstream gene network module. Therefore, to model this downstream module, one should be able to define a fixed profile for this pulsating gene.

This is what we have the 'fixedcomp' statement for: it defines an observed but unexplained, predefined course of a component's amount through time. It can be used if one has measured quantitative data for one component, and one investigates how hypothetically dependent other components react. The 'fixedcomp' statement defines a sequence of time-amount couples, for instance:

fixedcomp A 0 0, 2 50, 4 0, 10 0, 12 50, 14 0

would define a component with two peaks, at time points 2 and 12; and

fixedcomp B **repeat** 0 0, 2 50, 4 0, 10 0

defines a component showing an infinite series of repetitive peaks separated by 10 units.

Note that the above lies in the spirit of the observation that genetic networks appear to be highly modular (Hartwell 1999, Thieffry 1999). Therefore, as a way to start, such modules can be investigated relatively independently, but depending only on a few predefined input connections that can be defined via a 'fixedcomp' or perhaps via a mathcomp (see below).

Mathematically defined components

A so-called 'mathcomp' augments the functionality of both virtual and fixed components. It goes a lot further in fact, as it allows defining any mathematical function of other components and of time. For example:

mathcomp A = **cos**(**time***2***PI** / 40) + 1

would define a sinusoidal function oscillating between 1 and 0 with an interval of 40 time-units. Supported operations and functions are the common +, -, *, /, parentheses (), power ^, PI, time, sin, cos, exp, log, sqrt, abs, round.

Mathcomps are especially useful when SIM-plex is used to combine modelling scales, as will be evident in chapter 4. This means that not only components at the molecular level, but also components at the microscopic or macroscopic scale are all connected in the same model. As our research took place in the area of the plant cell cycle and how this is connected to leaf development, this type of cross-scale modelling extensions were an important feature of SIM-plex. A few examples:

mathcomp Ratio = CellArea/DNA

mathcomp LeafArea = CellNo * CellArea

where non-gene components like DNA, CellNo can be defined via "triggers" (section 3.4.2).

3.4.2 Triggers

As mentioned before, it can be interesting to build biomolecular networks that connect different levels of abstraction. Next to the basic genetic components, one may also want to model phenotypical or non-continuous parameters, which are still directly dependent on molecular causes. This is particularly useful for coupling the genetic network of the cell cycle with the physiological parameters of leaf development, like number of cells or cell mass.

For example for modelling the checkpoint of mitosis, it is necessary to let a component 'mass' slowly rise and to abruptly divide it by 2 when the checkpoint for mitosis is passed (cell division). This can be seen as a trigger that is fired every time a condition undergoes a transition from 'false' to 'true'. This conditional trigger would be defined here as "if the Mitosis-Promoting-Factor complex has accumulated sufficiently, and if the cell mass has risen above a certain minimum amount, then at that time the mass gets divided by 2". In a SIM-plex statement this goes like:

trigger if MPF>50 **and** mass>1.5 **then** mass = mass / 2

Another example that speaks to the imagination is the modelling of DNA accumulation during the process of endoreduplication (DNA redoubling without cell division).

trigger if MPF>50 **then** DNA = DNA * 2

Note that a trigger only fires when the condition passes from false to true, and will only fire again after the condition's value has first returned to 'false' again.

3.4.3 Multiplicative if-then

Another extension that allows crossing modelling scales is one that handles multiplicative conditional rates. In the original syntax, an if-then condition could only add or subtract a certain amount of gene product per time-unit. By introducing multiplicative if-thens, one can also multiply or divide the component amount with a certain factor during each time-unit. This SIM-plex extension was first at hand in the study of leaf cell surface area growth, which was modelled by increasing it each time-unit by a factor of the current size. As a statement:

if A>50 **then** cellArea * 1.05

Note that this can be seen as closely related to the degradation part already present in the PLDE formalism, but now with an incremental instead of decremental factor.

3.4.4 Gene regulation vs. protein reactions

This section should be considered as a more advanced topic. As mentioned in section 3.2.2-'Transformation', components used in a model may sometimes represent the total of gene product, meaning the RNA as well as the protein. This can be justified when the RNA is directly translated to proteins, and protein levels are reflected by mRNA amounts. At other times it may be necessary to sharply define the difference. In SIM-plex one can use both approaches. One can simple using two separate components (like *A_RNA* and *A_Protein*), or one can leave the RNA/protein distinction vague by just using one gene component. But then, in the latter case, consider the following fragment:

if X>10 **then block** A

if Y>10 **then transform** A_phosph **to** A 5

This piece defines that the gene *A* is transcriptionally blocked by *X*. But at the same time some *A* can still be created by dephosphorylation of a set-aside phosphorylated form of *A*. This means that the if-then-block statement should not block all of A's creation, but only the transcriptional part. And also, the if-then-transform part should work on the non-transcriptionally regulated level. Therefore, since SIM-plex can work with one level of gene-product, it has to work with two possible levels of regulation (only used when this appears to be necessary). One level is the transcriptional level of gene regulation and the other is the level of biochemical protein reactions, as shown in figure 3.5. In practice, this is taken care of by SIM-plex behind the screens, but sometimes one may want to define this explicitly. For this SIM-plex provides the 'nonreg' keyword (which stands for non-transcriptionally-regulated creation), for instance:

if A>10 **then** MPF **nonreg** 2.5

Note that if the 'nonreg' keyword is omitted, SIM-plex assumes that one is talking about a transcriptional-'block' regulatable component, or a component representing purely a protein.
Also note that instead of using the "nonreg" tail it would be preferred to use the more powerful if-then-transform statement (if applicable), which already makes use of the nonreg-functionality.

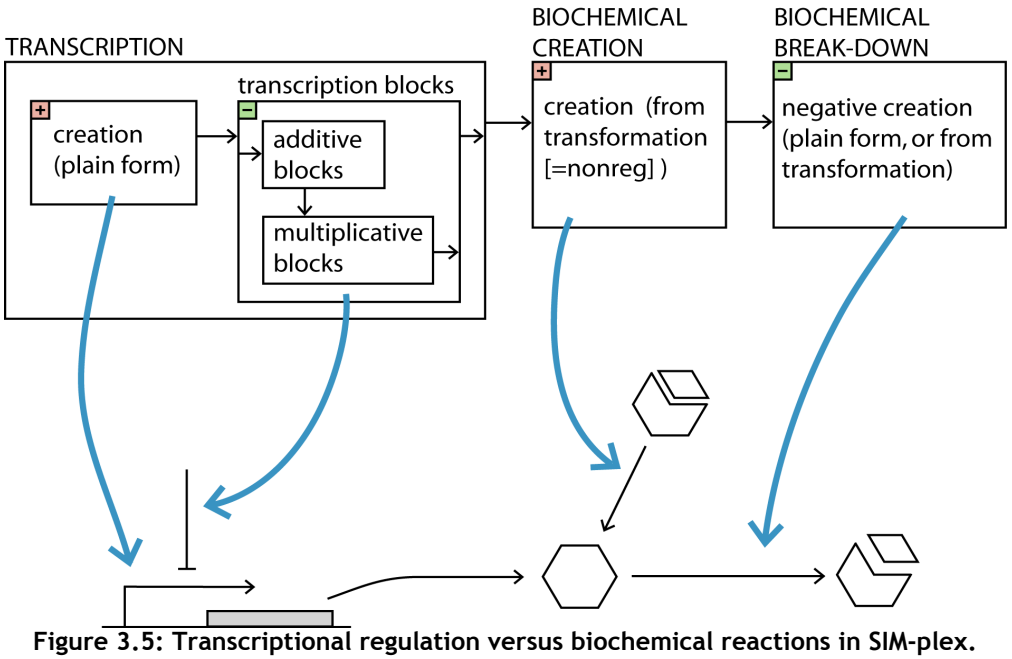


Figure 3.5: Transcriptional regulation versus biochemical reactions in SIM-plex.

To summarize, the difference between "... then A 5" and "... then A nonreg 5" is that the former models transcriptional creation which can be regulated by block statements, and the latter, "nonreg"-creation, can not be regulated by blocks. To give an overview: the if-then statement can have different tails that mean different things. Here is an overview:

(1) ... **then** A 5 = transcriptional creation;
(2) ... **then** A **block** 0.15 = blocks the transcriptional creation;
(3) ... **then** A **nonreg** 5 = non-regulated (un-'block'-able) biochemical creation;

Note how this issue nicely illustrates that things always tend to get a little more complicated than what appears from the surface. Luckily, this aspect of model definition will usually be of no burden to the user.

3.4.5 PLDE equation export

This functionality was included in order to support eventual translation of a model in the PLDE formalism into an equivalent model in the ODE formalism. When a user chooses this option via one of SIM-plex' menus, the program will combine the current statements and generate PLDEs for them, just as it would before a simulation is run. Then it exports these equations to a plain text file of choice.

A note about PLDE to ODE conversion.

As mentioned in section 2.2.5 and 2.2.6, and as seen on figure 2.6, full ODE equations use the Hill-curve to model gene activation. PLDEs differ from ODEs in that they approximate these activation details by replacing the Hill-equation by a step-function. So when one takes the PLDE equations exported by SIM-plex, one can simply replace the step-functions back to Hill-functions (and consider the extra parameter) to arrive back at full ODEs. This of course, given that no non-PLDE functionality like triggers etc. were defined. The resulting ODEs can be subsequently analyzed with existing mathematical tools for that formalism.

Measuring programming progress by lines of code is like measuring aircraft building progress by weight.
- Bill Gates

3.5 About the software

All the described statements plus all their options (see section 3.7 for the reference manual) have to be interpreted (so called *parsed*) by SIM-plex, in a manageable and moreover extendable manner. Therefore the parser module of SIM-plex uses the JavaCC (Java Compiler Compiler) package (https://javacc.dev.java.net). This package supports the creation of compiler software. It enables programmers to compose complex regular expressions (pattern definitions for text, which have to comply to a pattern themselves), and it allows to relatively cleanly insert programming code between regular expression parts, that will only be executed if that part is active. To give an idea, SIM-plex' parser alone counts 2000 lines of code of the current approximate 20'000 lines (including the parser, the simulator, the GUI, and an experimental parameter estimator not yet available in the released version). Admittedly, such a count only gives a quick impression; it doesn't address the complexity inside.

SIM-plex was published in the journal *Bioinformatics* (Vercruysse 2005), and is accompanied by an elaborate website that includes a tutorial and a detailed manual about SIM-plex' statements among others (http://www.psb.ugent.be/cbd/papers/sim-plex). There, one can also freely download the software. Note: if this URL would ever expire, try googling for "SIM-plex genetic network simulator" to find a new location.

3.6 Appendix A: Tutorial for new users

Instead of including this appendix here, we refer to the website (http://www.psb.ugent.be /cbd/papers/sim-plex) that accompanies the SIM-plex. The tutorial basically repeats the above explanation, but in a more compact, summarized form.

3.7 Appendix B: SIM-plex reference manual

As in the previous section, instead of including the entire reference manual here, we refer to the website (http://www.psb.ugent.be/cbd/papers/sim-plex) that accompanies the

SIM-plex software. In contrast to the exposition given in this thesis, it describes SIM-plex in a much more extended but also technical form. It includes:

- A full listing of SIM-plex statements, given in the form of regular expressions, plus documentation about their mathematical basis and intended use.
- Full description of more of the menus in the graphical user interface.
- A short note on the command-line interface to SIM-plex.

Chapter 4

SIM-plex: Application studies

4.1 Yeast Cell Cycle

As a validation that SIM-plex and the mathematical model of PLDEs are capable of modelling real biological networks, we first applied this to the cell cycle network of fission yeast (*Schizosaccharomyces pombe*). This gene network was already previously studied by Novak and Tyson (Novak 2001), but then with biochemical differential equations (ODEs). We wanted to check if SIM-plex could predict the same dynamical behaviour as was observed with ODEs.
Figure 4.1 shows the gene network as originally composed by Novak and Tyson as both a graphical sketch-up and as a set of differential equations. Next to that, the logical if-then SIM-plex statements are depicted. Figure 4.2 is a screenshot of SIM-plex after simulation of this network.

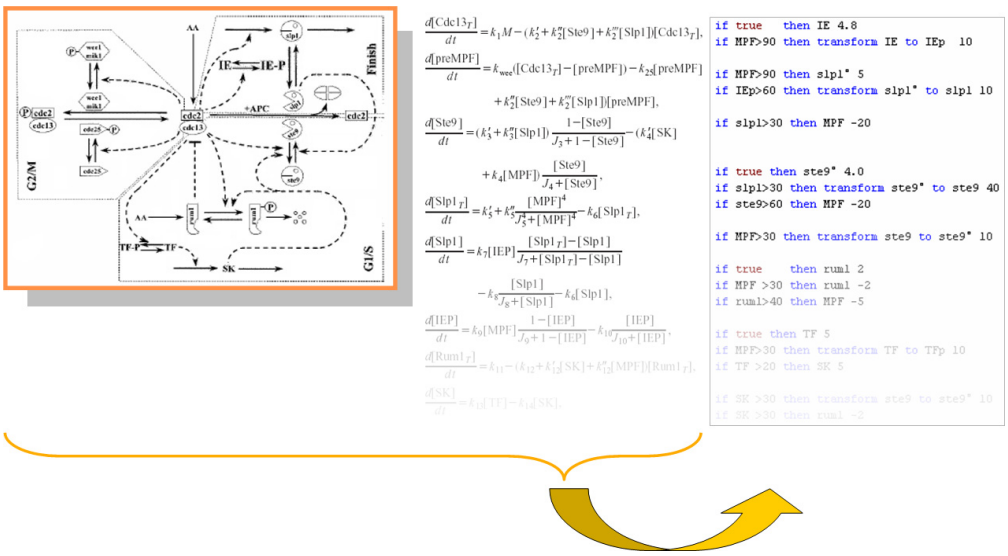


Figure 4.1: Novak's cell cycle network for fission yeast, and the translation into biochemical differential equations (ODEs) versus the translation in SIM-plex statements.

As shown in figure 4.2, SIM-plex predicts a cyclic behaviour based on the given components in the network, and we found that the timing and general gene product profiles match with Novak's simulation. For example the control of the MPF (Mitosis Promoting Factor) protein complex also shows the three levels of activity that represent the G1, G2 and M phases of the cell cycle. In figure 4.2, this is visible in the top of the single-plots window,

where MPF has a cyclic behaviour that stays 0 for a while (G1 phase, gap 1), then goes up (S phase or DNA-synthesis) and stays there for some time (G2 phase, gap 2), and finally undergoes a second boost in activity to trigger M phase (mitosis, cell division). Novak's publication itself used parameter estimations to match available profiles from experimental results. For our model, the parameters were estimated according to gene profiles in Novak's publication (timing delays and strength of activity), but this kind of fine-tuning was only possible since the structural basis of the behaviour was already in place.

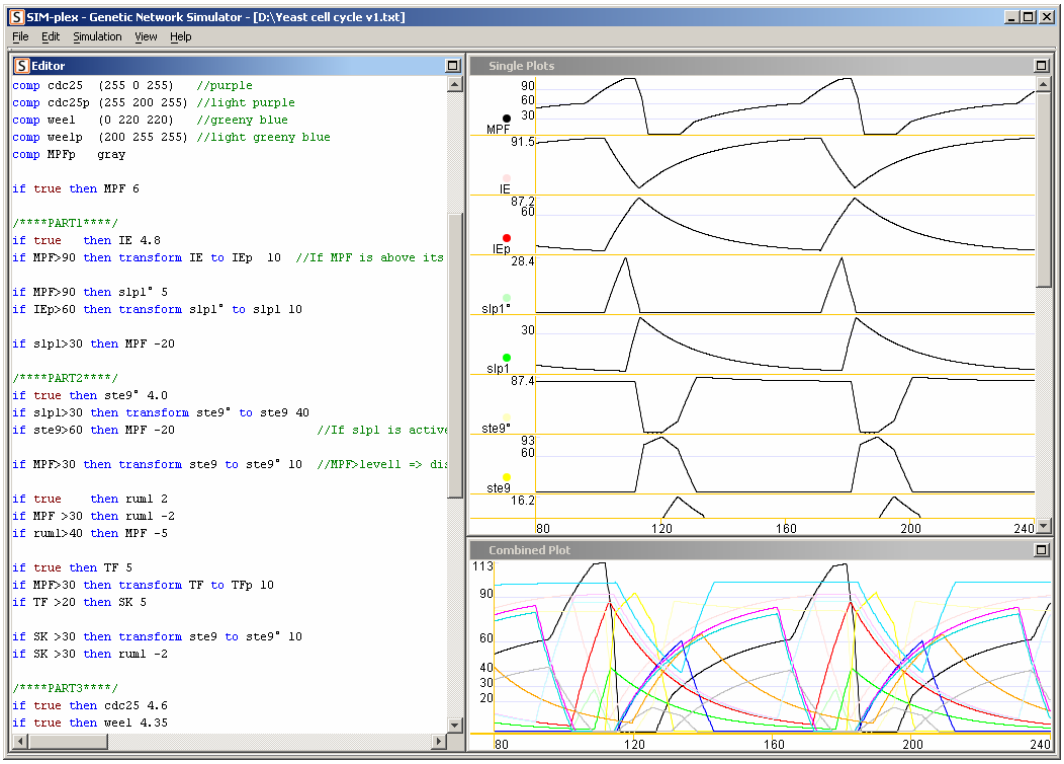


Figure 4.2: Simulation of the fission yeast cell cycle network in SIM-plex.

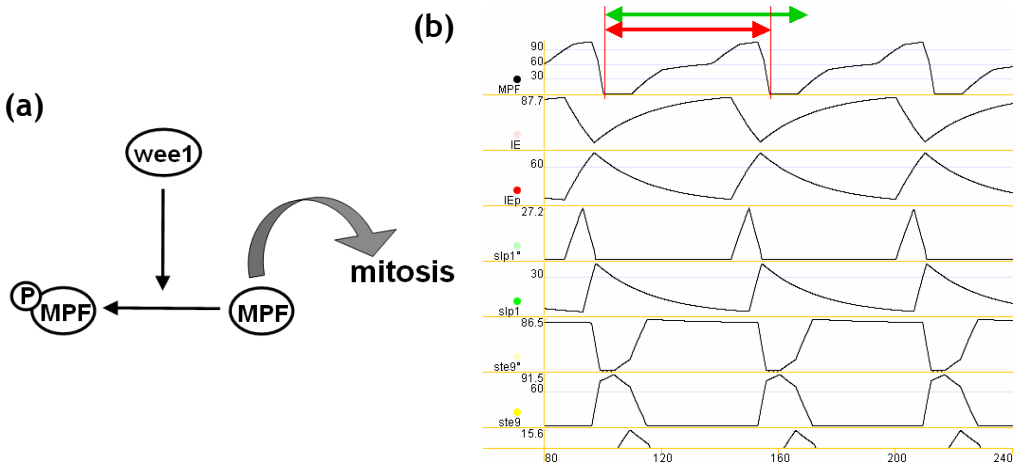


Figure 4.3: Model and simulation of the wee1- mutant. (a) Fragment of the fission yeast model, showing how active wee1 would normally inhibit mitosis. (b) Simulation of a model in which wee1 was significantly weakened. The upper arrows show the length of the original cell cycle duration, while the lower arrows show the shortened length in the wee1- mutant.

In further testing we ran a number of simulations on disruptions of the original network. For example we tested if the down-regulation of the G2/M-transition controller *wee1* would lead to a shorter cell cycle (because without wee1, the G2/M checkpoint could be passed quicker). This was already experimentally observed in wee1- mutants, as well as in Novak's fission yeast models. Figure 4.3 shows that a simulation in SIM-plex also indicates this fact: the upper arrow shows, as a comparison, the larger length of the cell cycle in the unmodified yeast strain. By comparing figure 4.3 with 4.2, one can clearly see in the profile that the control of the G2/M transition by wee1 is diminished: the cells stay less long in G2 phase (middle MPF activity level) and enter M phase more quickly. Analogously (not depicted here), the simulation of a rum1Δ ste9Δ yeast cell line also showed a shorter cell cycle, by loss of control at the G1/S transition (no more MPF level-0 period).

4.2 KRP2 role in Arabidopsis endocycle

Biological background

As a first application in current molecular genetics research, SIM-plex was applied in the investigation of the role of KRP2 in the cell cycle of *Arabidopsis thaliana* (thale cress, or 'zandraket' in Dutch). *Arabidopsis* is a favourite biological model system because it has a small generation time (compared to e.g. poplar trees) and a small genome size (115 Mbp or mega-basepairs, compared to rice with 430Mbp, or maize with 2500Mbp, or wheat with 15000Mbp). Still, the fundamental, evolutionary well-conserved cell cycle process and its genetic regulatory programs are very comparable to many other plants or even species in general, and can be used to extrapolate genetic knowledge across the various species.
The research we describe here focuses on how the KRP2 gene/protein influences the transition from normal mitotic cell cycle to a shortened, endoreduplication cycle. This so-called endocycle still goes through G1 and S phase, but it skips the M phase or mitosis. The cells don't divide anymore but the DNA content doubles during each cycle. Since endoreduplication coincides with dramatic cell expansion in leaf cells, this suggests that endoreduplication could be the driver of growth (Beemster 2005), or is at least tightly connected to it. So gathering thorough knowledge about the endoreduplication control mechanism may be vital to understand the molecular basis of growth and biomass production.

Components and interactions

The publication that resulted from this study (Verkest 2005) elucidates the interaction between three key components in the endocycle: KRP2, CDKA;1 and CDKB1;1. Through arduous wet-lab work of one-by-one genetics, a number of regulatory interactions between those components could be postulated, see figure 4.4. Subsequently, we ran a number of dynamic simulations for normal and modified KRP2 or CDKB1;1 levels (transgenic plant lines), to illustrate and to further support the validity of these interactions. From a systems biological viewpoint, however, the gene network fragment studied here is a modest one. But still, it helps us researchers to think about the dynamics of the system. This became apparent since after the publication was finished, the model was used as a basis for several more extensions and for testing hypotheses about connectivity.

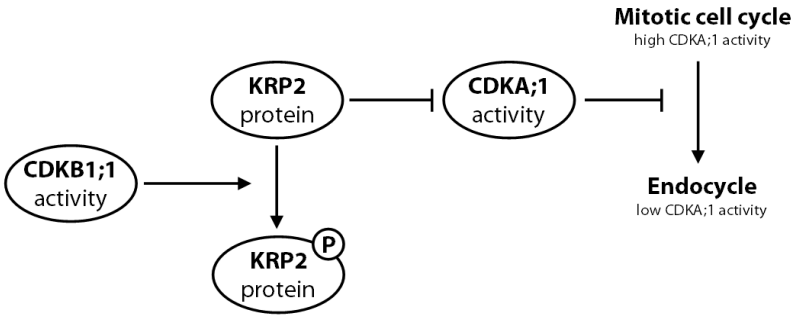


Figure 4.4: Model illustrating the role of CDK and KRP activity in controlling the onset of endoreduplication. At sufficient activity of the CDKB1;1 complex, it initiates KRP2 protein phosphorylation. This phosphorylation serves as a marker for subsequent quick destruction, which is thus a mechanism to reduce KRP2 levels. KRP2 is an inhibitor of CDKA;1 complexes (bound to several types of cyclins), which in turn supports the mitotic cell divisions.

The model

We first predefined the activity of CDKB1;1 through time because the experiments focused only on downstream CDKB1;1 interactions. We defined its profile as spikes between approximately every 11th and 15th hour of a 20 hour long mitotic Arabidopsis cell cycle, as observed by (Menges 2002). The KRP2 protein was defined to be continuously produced, only to be phosphorylated (and then degraded) when CDKB1;1 activity exceeds a critical threshold. When KRP2 reaches its own critical threshold, it inhibits CDKA;1 activity. A strong CDKA;1 activity is held responsible for keeping the cell cycle in a mitotic state; a weaker activity makes it change states to the endocycle; and non-activity would completely stop the cell cycle.

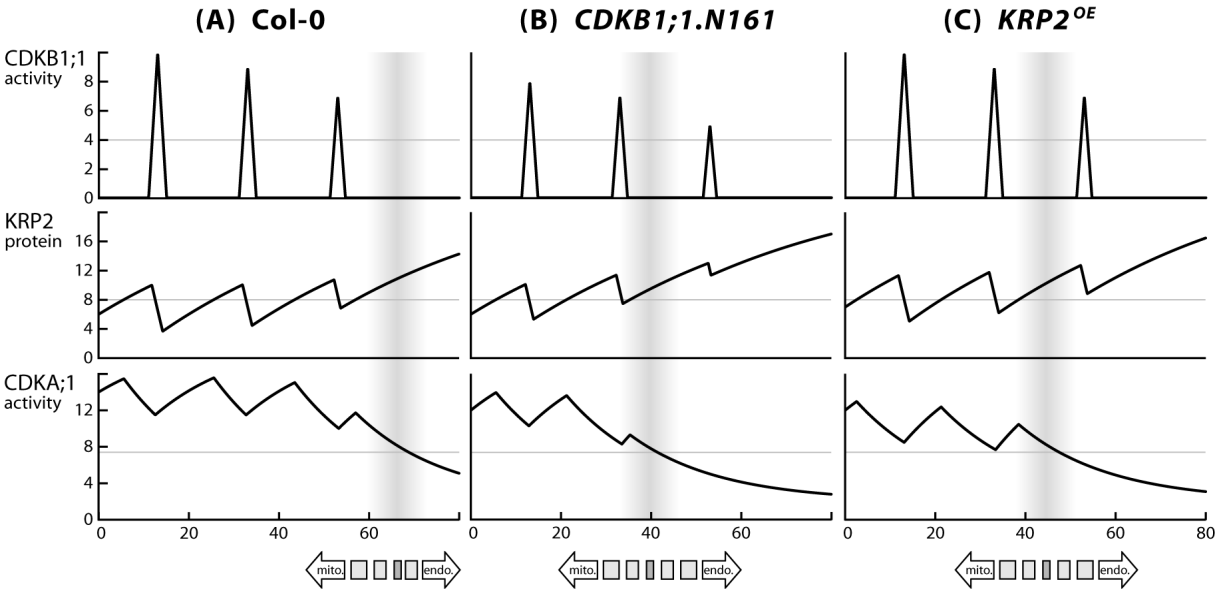


Figure 4.5: Simulations of control of the mitosis-to-endocycle transition.
The simulations were set to run over four times the duration of a normal mitotic cell division cycle (4 x 20h), and show KRP2 protein abundance and CDKA;1 activity based on predefined CDKB1;1 activity. Activities were simulated in arbitrary units.
CDKB1;1 initiates KRP2 phosphorylation when it reaches the activity threshold of 4 units. Next, KRP2 inhibits CDKA;1 activity above its threshold 8. CDKA;1 activity first remains above a level that is assumed to be necessary to maintain mitotic division. In the grey-colored zone, CDKA;1's mitotic activity drops below this level, marking the onset of the endocycle. Note the remaining CDKA;1 activity at this stage to keep the endocycle running.
Simulations are for a wild type and two modified plants: (a) wild-type (Col-0) plants; (b) dominant negative CDKB1;1 overexpressing plants; (c) KRP2 overexpressing plants.

Simulations

The simulation for wild-type plants (Figure 4.5a) shows that as long as the KRP2 activity remains controlled, the CDKA;1 level makes relatively small oscillations through the cell cycle, but its activity remains high enough to enable mitotic divisions. Only when CDKB1;1 does not any longer attenuate KRP2 sufficiently, KRP2 amounts rise enough to push the CDKA;1 activity below the level needed to keep the mitotic machinery active. Note that the activity is not pushed to zero as there still remains activity to keep the endocycle running.

Two additional simulations were performed, modelling two transgenic lines that enter the endocycle early compared to the wild-type. The first represents a mutant overexpressing a dominant negative CDKB1;1 allele (Figure 4.5b). The simulation shows that although KRP2 activity starts at the same level as in wild-type, it receives less negative control (CDKB1;1 has a smaller period of activity) and rises more quickly, resulting in an early deactivation of mitotic CDKA;1 activity and an early endocycle entry. The second alternate simulation models a KRP2 overexpressing line (Figure 4.5c). The CDKB1;1 levels are the same as in wild-type and they phosphorylate about an equal amount of KRP2 protein as in wild-type (note the size of KRP2's decrease in the three simulations). But now there is a higher quantity of active KRP2 protein present in the cells to keep the activity high enough and to strongly diminish the CDKA;1 activity. The two transgenic line simulations show an early endocycle entrance, which is consistent with what is seen in the experiments.

The capacity to mimic true *in vivo* situations through dynamical modelling allows us to hypothesize that the interactions as presented in Figure 4.4 are an important part of the core mechanism controlling the mitosis-to-endocycle transition during Arabidopsis leaf development.

Network definitions

This is the SIM-plex statement list used to define the wild type model:

```
fixedcomp CDKB11 0 0, 11 0, 13 10, 15 0,        //peak 1
     31.1 0, 33 9, 34.9 0, 51.3 0, 53 7, 54.7 0 //peak 2 & 3
comp KRP2   6  0.02
comp CDKA1 14
comp KRP2ph

timepoints 0 to 80

if true then KRP2 0.5
if CDKB11 > 4 then transform KRP2 to KRP2ph 3

if true then CDKA1 1
if KRP2 > 8 then CDKA1 -0.9
```

Network definition of the CDKB1;1.N161 line differed from the wild-type only by a lowering and shortening of the CDKB1;1 activity: "fixedcomp CDKB11 0 0,11.2 0,13 8,14.8 0,31.3 0,33 7,34.7 0,51.5 0,53 5,54.5 0", and by reducing the initial level of CDKA;1 (12 instead of 14), because of a history of higher KRP2 activity prior to time=0. Network definition of the KRP2 overexpressing line only differs from wild-type in KRP2 initial level (7 instead of 6), KRP2 continuous creation rate (0.55 instead of 0.5) and initial level of CDKA;1 (again 12). The simulation results were exported into Adobe Illustrator for optimal presentation in

figure 4.5. Full network definition files and SIM-plex screenshots are available as additional data at the URL: http://www.psb.ugent.be/cbd/papers/krp2sim.

4.3 Lateral root development: the auxin switch

Biological background

The plant hormone auxin is a strong stimulator of cell division. In order to better understand the molecular details of this auxin signalling pathway and its connection to the core cell cycle, Vanneste et al. have chosen to study lateral root development. This biological model system is used to investigate the auxin/indole-3-acetic acid (AUX/IAA) signalling pathway and how it regulates the cell cycle downstream (Vanneste 2005). As an exercise, we took their results, the molecular components and interactions they proposed, and we were able to show how dynamical simulations can reveal non-obvious behaviour of the system.

The modelled network: high-level view

The proposed model is shown in figure 4.6. The model postulates that lateral root initiation happens under the influence of two counteracting parts of a regulatory circuit. When the auxin concentration is low, the circuit keeps small auxin increases under control, like by binding it to other biomolecules. However, if a strong influx of auxin finds its way into the cell, then this negative regulatory part is overruled. Instead then a positive regulatory part becomes active, which now helps to sustain the high auxin levels while the developmental program of lateral root initiation is being activated.
Basically, for low auxin amounts, one part of the network is able to keep the concentration of auxin under control and maintain homeostasis. But for high auxin amounts, the other part of the network takes over by inhibiting the first part, and by reinforcing stable, high auxin levels. This essentially works like a binary switch in auxin activity.

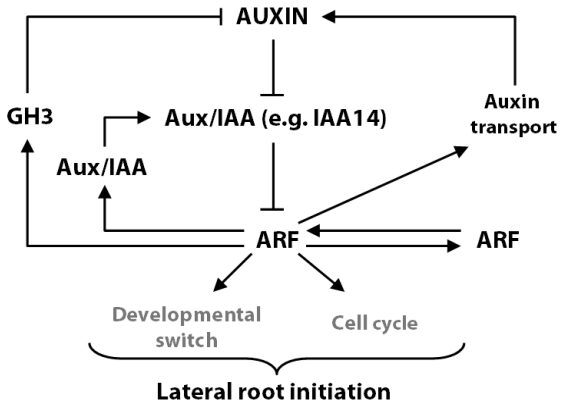


Figure 4.6: The network of auxin-induced lateral root initiation, according to Vanneste et al. See the main text for a detailed component interaction description.

The modelled network: components and their interactions

Auxin promotes the degradation of Aux/IAA proteins that prevent auxin response factors (ARFs) to regulate auxin-responsive target genes. Aux/IAAs and ARFs represent large gene families in Arabidopsis (Weijers 2005). The negative feedback loop of this model (figure 4.6, left half) is itself made of two parts. First there is ARF that stimulates Aux/IAA protein

formation, which by capturing its activator ARF again, inactivates ARF. And second is the formation of GH3, which encodes for auxin-conjugating enzymes that inhibit auxin accumulation. The positive feedback loop of the model (figure 4.6, right half) consists of ARFs that positively enforce each others' creation, and of an ARF-mediated activation of auxin transport, hereby directing higher levels of auxin into the cells.

Simulation dynamics: low auxin

The complete network definition is visible in the left hand side panel of the SIM-plex screenshot in figure 4.7a. On the right hand side appear the simulation result plots. These show a rather complex profile, which we will now describe in detail. One can follow this description on the figure.

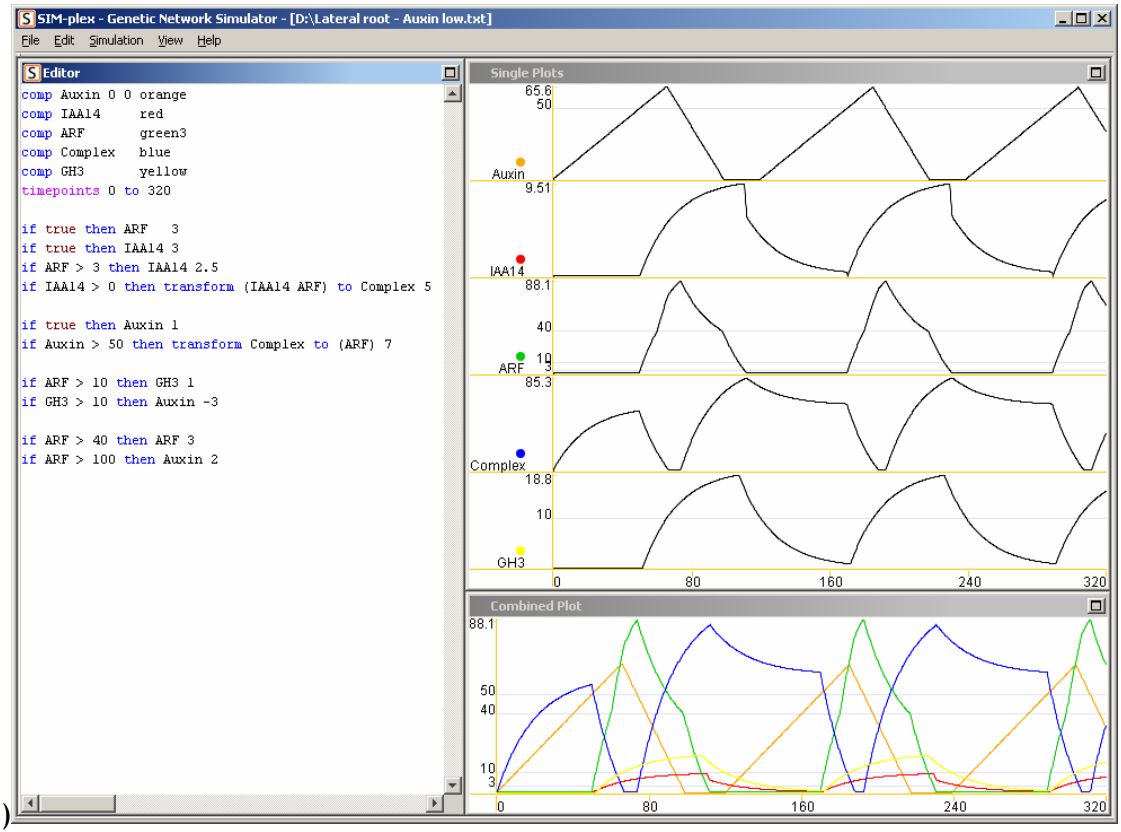
To kick-start the system, we assessed the requirement that ARF and (Aux/)IAA14 are being created continuously, although by a yet unspecified cause. All these ARF and IAA14 then immediately bind, forming a component called 'Complex'. In the mean time, auxin also rises, until it reaches a critical level of 50 units. There it starts decoupling the inhibitory IAA14 from the ARFs, after which the IAA14s are destructed. Complex dissociation is now stronger than Complex formation, and ARF is accumulating. Reaching a level of 10 units, ARF now stimulates GH3. Meanwhile, ARF abundance still gets a boost from its ARF partners that are 'waking up' as an auxin response. But, then GH3 gets active as it reaches a level of 10 units. As a result, auxin levels start falling, and soon they drop below 50 and cannot sustain ARF liberation from IAA14 any more. The bound Complex of ARF+IAA14 starts rising again, pure ARF levels drop, and GH3 loses its activator. When GH3 subsequently loses its auxin-inhibiting power, auxin levels can rise again, and the loop will repeat. The result is a cyclical series of events that ultimately keep the auxin concentration between reasonably low bounds.

Simulation dynamics: high auxin influx

The second model's sole difference from the first is the initial amount of auxin. In the first simulation, it is low (zero for a start), and in the second, it is high (500 units). This single change now models the situation under higher auxin influx into the cell, and has a dramatic effect on the dynamics of the system, see figure 4.7b.

Basically, compared to the previous description, auxin levels will now only drop a little, but not enough to deactivate it before it has triggered ARF levels to reach critical mass. At this point of 100 units, ARF can enhance auxin by stimulating it as well. The final result is near-zero Complex levels (mind the scale of this component in the simulation plots), moderate Aux/IAA14 and GH3 levels but not enough to deactivate auxin (yet), and strong Auxin and ARF levels (ARFs, which are connected to cell divisions).

(a)



(b)

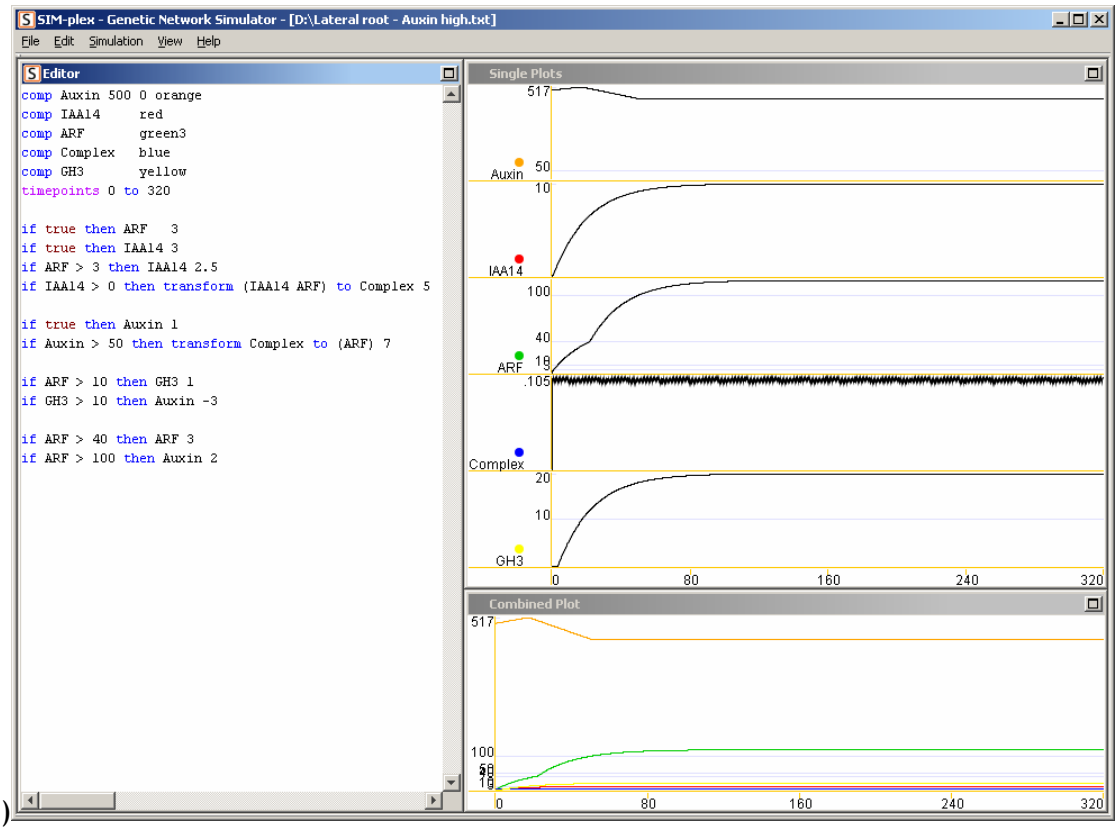


Figure 4.7: Simulations in SIM-plex of auxin-induced lateral root initiation. Part (a) shows the oscillating, auxin-dampening behaviour under low auxin influx, while (b) shows the auxin-reinforcing behaviour of the same network under high auxin influx.

4.4 Basic Arabidopsis Cell Cycle in leaf development regulation

Biological background

Leaf development is the process programmed in plants to make a small green bud grow into a mature, structured leaf. Leaf development is used in molecular biology as a model system to study the cell cycle, as it happens in three rather distinct phases that directly reflect the three phases of cell cycle steering (Beemster 2006). The first phase is the proliferation phase, where cells grow and divide, while repetitively going through S (DNA-copy synthesis) and M (mitotic) phases. In the second phase, called the expansion phase, cells have stopped dividing but now they expand dramatically and the leaf grows considerably in size. At this time, cells are skipping the M phase and the cell cycle has switched to endoreduplication mode. Leaf growth is now the result of cell expansion. The DNA is just being replicated repeatedly, with levels up to 32 times the normal DNA amount or even more is no exception. The third phase is maturity, where no more division or growth occurs. One should still consider the presence of some overlap between the phases during the development; for example one observes a tip-to-base gradient during the phase transitions.
Note that leaf development is not only a model system to gather knowledge about the cell cycle, but is also an interesting area connected to biomass production, which has economical and industrial implications.

From data to model

In our publication (Beemster 2006), we set out to establish a first link between the higher levels of leaf growth and cellular status, and the molecular level of gene activity that drives the cell cycle. Kinematic growth analysis results for the abaxial leaf epidermis were used for the first pair of leaves of Arabidopsis (Beemster 2005). This consisted of data on total leaf surface and average cell size, and allowed to calculate average cell expansion rate and cell division rate as functions of time. Furthermore, flow cytometry was used to measure DNA ploidy (the DNA content of cells), which allows to discern between mitotic divisions versus numerous rounds of endoreduplication. Finally, microarray data yielded gene expression profiles that gave transcriptional footprints of the genome during the various stages of leaf development. Combining the results, it was found that the different observable stages of leaf development (proliferation, expansion, and maturity) were clearly reflected in the molecular footprint, thus pointing out an obvious cross-scale but also temporal relationship. Next, in order to formalize assumptions about a connection between growth and cell cycle regulation, we built a simple model in SIM-plex. This way, we could simulate the dynamical behaviour and compare in how far its implied phenotypic predictions correspond to the experimental facts.

Model and simulation

The model proposed here (see figure 4.8a) is driven by a growth factor component that flows in through the base of the leaf and decreases in availability over time, due to the ongoing growth of the leaf. As it causes cells to grow (modelled by a CellArea increase), cell growth will also diminish when the growth factor concentration drops. A critical ratio between cell size (area) and DNA content triggers the production of CyclinDs, which bind

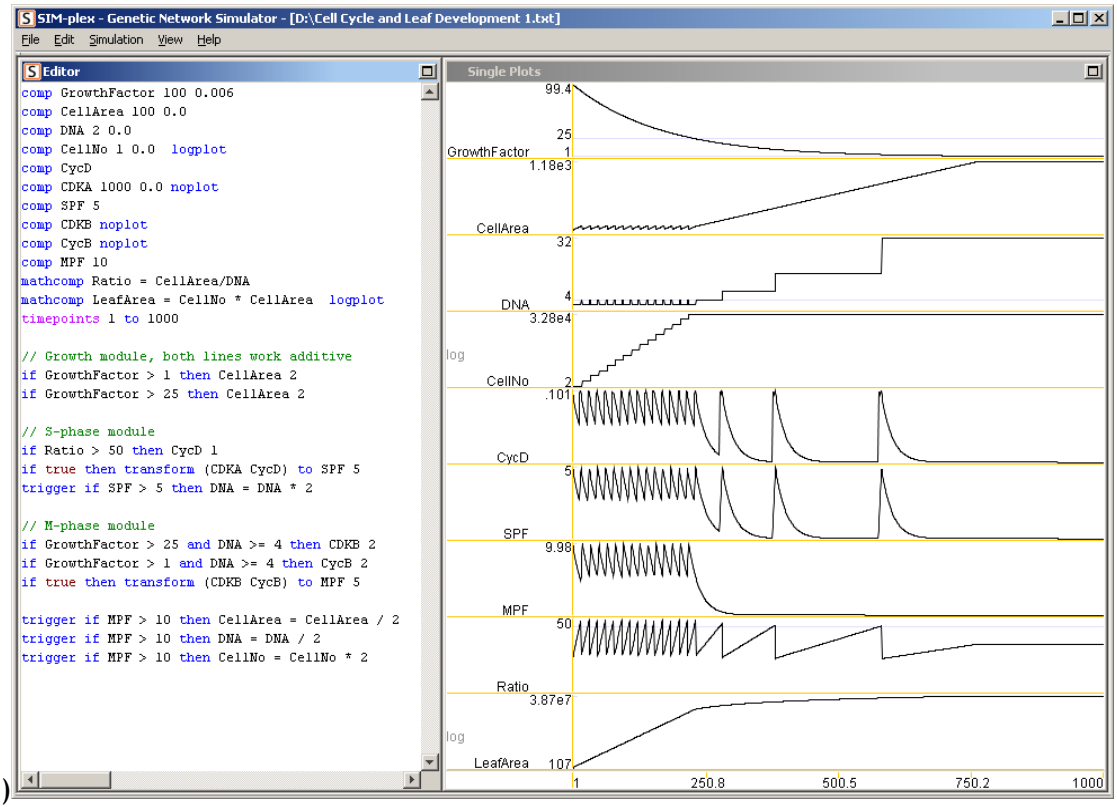
with CDKAs and give rise to the S-phase promoting factor (SPF). This will after a while cause the DNA to duplicate. After this, CyclinBs and CDKBs start forming an M-phase promoting factor (MPF), which causes cell division. Although the model represents the molecular situation in one cell, at the same time a cell count is kept for the entire leaf, which increases at each cell division event. Also, the CellArea halves, and from these two factors the total leaf size is calculated. As a result, indirectly through the three different GrowthFactor levels, the simulation shows the three modes of the cell cycle, corresponding to the three modes of leaf development. During the proliferation phase, cell number and leaf area increase exponentially. During the expansion phase, only leaf area continues to increase, although at a slower rate; and also endoreduplication occurs, as can be seen in figure 4.8a as a stepwise increase in DNA content.

Inhibited cell cycle indications

A small modification to the original model is made in figure 4.8b. Here the cell cycle progression is slowed down by the inhibitor KRP2, what is modelled by simply lifting the activation thresholds for both SPF and MPF. As described in more detail in our publication (Beemster 2006), a comparison between the two models' simulation results leads to a number of interesting observations. Firstly, linear cell growth fits well for the proliferation phase. But for the expansion phase, however, cell growth appears to happen exponentially. This became apparent from the fact that, if cell growth is modelled as linear (via core SIM-plex statements), the simulation predicts quasi-equal final cell sizes in wild-type and $KRP2^{OE}$, whereas in reality, their final cell sizes differ. Note that it is difficult to measure individual cell growth in the leaf system. Subsequently, biological hypotheses were formulated to explain this difference in growth behaviour. In this respect, the model pointed out and supported new insights into the mechanism involving the relationship between cell cycle regulation, cell expansion and whole organ growth. This model was developed by (Beemster 2006) and was made possible by the new crossing-scales extensions that were introduced into SIM-plex.

Different phases: | Prolif. | Expansion | Maturity |

(a)



(b)

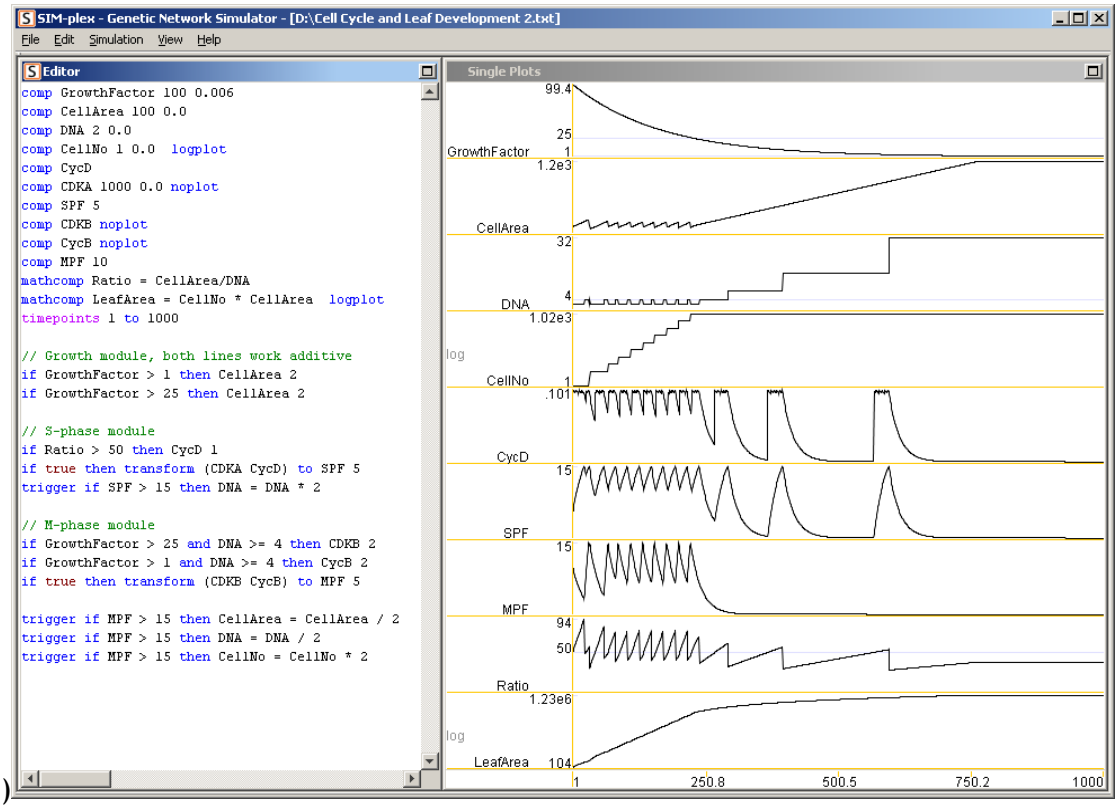


Figure 4.8: Model definitions and simulations in SIM-plex of the connection between cell cycle control and leaf growth. (a) shows the wild-type situation, while (b) illustrates overexpression of an inhibitor KRP2 via increased DNA duplication and mitosis threshold values. Note that the components *CellNo* and *LeafArea* have a logarithmic scale on the vertical axis, while the other components have a linear scale. Note also that the scale for cell number is significantly different in the two graphs. Cell cycle inhibition decreases cell accumulation but does not modify mitotic exit.

4.5 Arabidopsis Cell Cycle

In order to gain a dynamical, Systems Biological understanding of the plant cell cycle, the aforementioned Arabidopsis models have been extended with several extra genetic components. Firstly, De Veylder (unpublished results) could expand the endocycle onset model of (see section 4.2) to 8 components (including CCS52A, CCS52B, E2Fa, and DEL1), so as to establish firmer systemic insight in the mitosis-to-endocycle control program. Secondly, I built a model with 14 components based of the most commonly available knowledge on the plant cell cycle, coming from (Inzé 2005) and personal communications. Omitting the molecular details, figure 4.9 diagrammatically sketches this model.

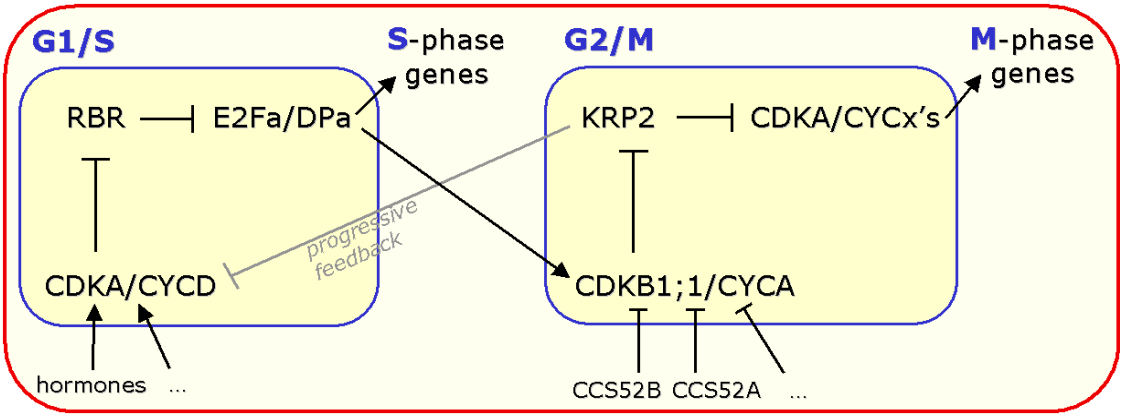


Figure 4.9: Diagram of plant cell cycle control. The two sections indicate the two different checkpoint control mechanisms of the cell cycle: S-phase entry (checkpoints like sufficient growth and nutrients), and M-phase entry (checkpoints like DNA replication complete). Details of the interactions are described in the main text.

The model assumes a repeated cell cycle initiation signal. In the yeast model this was signalled by the end of the M-phase but here it is a gene with cyclical bursts of activity like in the original KRP2 model. It could be activated under the influence of growth factors like hormones that stimulate the production of CyclinD. The G1/S transition control happens by CDKA complexes that inhibit the inhibitors of S-phase regulating genes. Eventually, these also lead to the startup of the G2/M transition, further pushed by CDKB complexes that inhibit the inhibitors of M-phase regulating genes. In order to establish the transition of mitotic divisions to the endocycle, KRP2 gains strength over time so as to inhibit the M-phase activating power of CDKAs. In the end, even the S-phase promoting CDKA complexes are affected so that the cell cycle ends completely. KRP2 gradually gets this attenuation power because its inhibitor CDKB is inhibited itself through the CCS52s. Below is given the statement list that defines the model, that gives more molecular detail, and that can be entered in SIM-plex to plot the dynamical behaviour:

```
fixedcomp CCstarter repeat 0 0, 0.1 10, 4 10, 4.1 0, 20 0
comp CYCD 0  noplot
comp CDKA 100
comp CDKA_CYCD 0
comp RB   0  0.05  noplot
comp RBph 0  0.05  noplot
```

```
comp E2FaDPa_RB 19
comp E2FaDPa 0
fixedcomp CCS52B 0 0, repeat 11 0, 13 10, 15 0, 31 0
comp CDKB1 0 0.02
comp DEL1 10 0.02
comp CCS52A
comp KRP2 4 0.02
comp KRP2ph 0 0.5 noplot
timepoints 0 to 160

//Cyclin D activation/deactivation
if CCstarter > 5 then CYCD 4
if CCstarter < 5 then CYCD -2

//Forming/breaking of Cyclin D + CDKA complex
if true then CDKA 5
if true then transform (CDKA CYCD) to  CDKA_CYCD  10
if true then transform  CDKA_CYCD  to (CDKA CYCD)  2

//Creation and automatic dephosphorylation of RB.
if true then RB 1
if true then transform RBph to RB 1

//Regulation of E2Fa/DPa
if true then E2FaDPa 1
if true then transform (E2FaDPa RB) to E2FaDPa_RB 10
if CDKA_CYCD > 9 then transform E2FaDPa_RB to (E2FaDPa RBph) 10

//Regulation of CDKB1 by E2FaDPa, CCS52B, and CCS52A (via DEL1)
if E2FaDPa > 8 then CDKB1  1
if CCS52B  > 5 then CDKB1 -5
if DEL1    < 5 then CCS52A 1
if CCS52A > 10 then CDKB1 -5

//Regulation of KRP2
if true then KRP2 0.5
if CDKB1 > 4 then transform KRP2 to KRP2ph 3

//Dose-dependent regulation by KRP2
if KRP2 > 5  then CDKA -1.5
if KRP2 > 18 then CDKA -4
```

Thirdly, Beemster (unpublished results) enlarged the cross-scale model of section 4.4 to a stunning 34 components. This includes 26 molecular components, among them different cyclins, E2Fa, E2Fc, DPa, CDKD, CAK, etc. This count again includes phosphorylation variants and protein complexes, as these have different properties and are should thus be considered as different components.

In conclusion, several successful modelling approaches have been carried out to gain a dynamical understanding of the plant cell cycle, in the model plant *Arabidopsis*. In order to further study this biological process, two tracks can be followed simultaneously. First of all, continued wet-lab research focused on the cell cycle will keep giving new clues about interactors. Second and not least, literature information has to be harvested so as to collect the many different but scattered clues about components, known or hypothetically related to the cell cycle. In fact, as the latter has proven to be a bottleneck for further modelling efforts, we have set out to develop new tools that help to assist the gathering of

existing but hidden information from literature. This will be the topic of chapter 5 and the rest of this thesis.

4.6 Follow-up: translation to full ODEs

The emergence of initial SIM-plex versions of *Arabidopsis* cell cycle models was possible because of the low threshold between biologists and this newly developed simulation software. As a next step, this ground-breaking work on an *Arabidopsis* cell cycle model is now being continued (in cooperation with the Mathematics department of Gent University) by translating it to the full Ordinary Differential Equation framework. Here, more analyses can be performed, making use of the full range of mathematical tools available for ODE analysis. For example phase space or stability analyses can be performed with MATLAB (http://www.mathworks.com) or MathGrapher (http://www.mathgrapher.com).

Part 2 -
Biological Information Management

In the middle of difficulty lies opportunity.
- Albert Einstein

Chapter 5

Status of information extraction techniques for biomedical text

5.1 The current biomedical information bomb

The biomedical sciences are witnessing an accelerating information explosion. Each year, over 600,000 new scientific publications are appearing, according to PubMed and Medline, the indexing authorities in the field (see figure 5.1). This observation puts forward the question of how to manage or oversee this jungle of information. For sure, one human researcher on his own can no longer read all these articles. Even when confining to a specific topic like the cell cycle, it is nowadays no longer possible to read all the new papers that come out each year (Jensen 2006). As for computer-assistance, the format of unstructured text of biomedical journal articles makes them largely inaccessible to computational methods. This paradox of vast knowledge production efforts but no method to efficiently cope with the results, demands for bold new solutions.

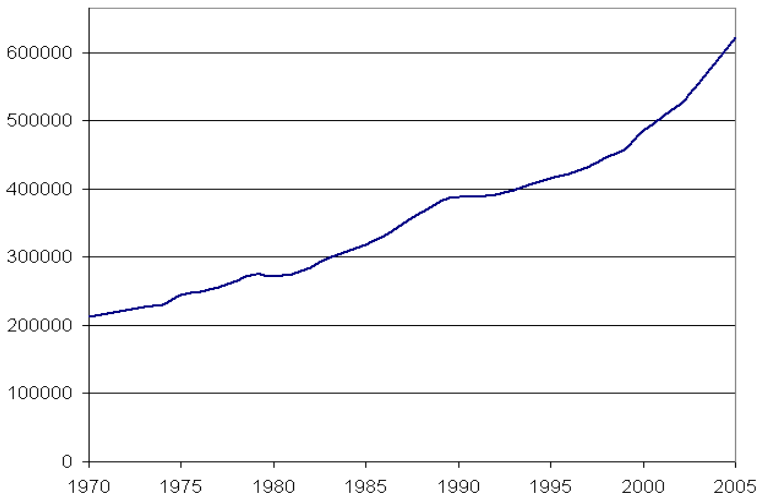


Figure 5.1 Number of new biomedical publications each year, as indexed by Medline. In 2005, (later years are not completely indexed yet), the barrier of a staggering 600,000 new publications per year was crossed (Ref.: 'Medline Citation Counts'). Medline and its interface PubMed are currently indexing over 17 million biomedical publications (Wheeler 2008).

Systems Biology as a driver for knowledge integration.

Also, with the advent of the Systems Biology paradigm, now more than ever an overview of this tremendously dispersed information is required. In this paradigm, researchers are trying to understand biological *systems* as an *entirety* of the elements that were previously discovered separately. These elements and relations should now be recovered and combined or *integrated* into one holistic view. As Kitano says: "In order to understand

biology at the systems level, we must examine the structure and dynamics of cellular and organismal function, rather than the characteristics of isolated parts of a cell or organism" (Kitano 2002). One sometimes compares this aspect of Systems Biology to figuring out how the distinct parts of a car ought to work together. Although one may have a clue about many genes' workings, in many cases one is still trying to expose the detailed role they all play in a composed machinery with tens or hundreds of other genes.
But in order to combine the pieces, one must first have them. So the first task is to collect and manage all the required pieces of information. To cite Erhardt et al (Erhardt 2006): "Effective knowledge management will be a key element for the success of the biotechnology and pharmaceutical industry in the years to come. Independent of the problem under study, revision and exploration of the knowledge already acquired is necessary for every researcher".

Institutional databases are not enough

Although many research results already find their way into data repositories dedicated to specific information types (Gene Ontology, KEGG, Reactome, etc), scientific literature can still cover a much broader area of knowledge, because natural language has so much more expression power. In addition, the sheer amount of new publications makes it exceedingly hard to keep these manually curated information resources up-to-date. Baumgartner et al. have even presented the numeric argument that this is simply not feasible (Baumgartner 2007). For example, they show that the number of studied mouse proteins that are missing a Gene Ontology (GO) annotation, is growing faster than the number of mouse proteins that have gained at least one GO annotation. The same observation is made for Swiss-Prot proteins, as well as for other organisms. These patterns lead to the conclusion that *institutional* manual curation processes will take far too long to complete the annotation of even only the most important model organisms, and that at their current rate of production, they will barely be sufficient for completing the annotation of all currently available and future biological data. Note that, despite the referred publication's quite general title, we have to stress that they only highlight the unfeasibility of manual text curation efforts, when executed by a confined number of people.

Automated literature mining is not satisfying either

Clearly, the biomedical field needs a mass approach to counter this information flood. Automated text mining techniques, where computer algorithms interpret the written text, may seem the obvious candidate to alleviate the problem. However, as we will show in the following sections, these automated techniques are still very far from optimal. Natural human language, certainly in expert fields like biology and medicine, is far too complex for present-day algorithms to grasp. The software makes too many errors against intended meanings, and misses out on too much essential information (Dickman 2003, Jensen 2006). Again, this leads to many missed opportunities of computer assistance and knowledge deduction. So, just like the limited-scale manual text curation mentioned before, large-scale computerized sweeps also only provide a partial contribution to the solution, and will not provide a definitive remedy in the near future either.

Before we then go on and propose our own, inevitable but logical solution, let's first have a more detailed overview of the current approaches. The existing efforts tailored to try and tackle the informational overload will give us an initial background to start from, and give

some perspective on the status in the field of information retrieval from biomedical literature.

5.2 Current automated solutions for literature harvesting

In order to make computers identify meaning in biological text, current algorithmic setups use the combination of two methods. The first step is the recognition of entities or terms, being words or clauses. The second step is the extraction of information by linking up these entities and finding out relationships between them. The following two subsections illustrate these two aspects.

5.2.1 Task 1: Entity Recognition: identifying the substance(s)

The goal of Entity Recognition is to uniquely identify all the terms that appear in a fragment of text, be it gene names, protein names, processes, organs, species or relationship types. For that purpose, first the words or the groups of words are identified that refer to entities, and second, these entities are mapped onto identifiers like a gene accession number or a dictionary identifier.

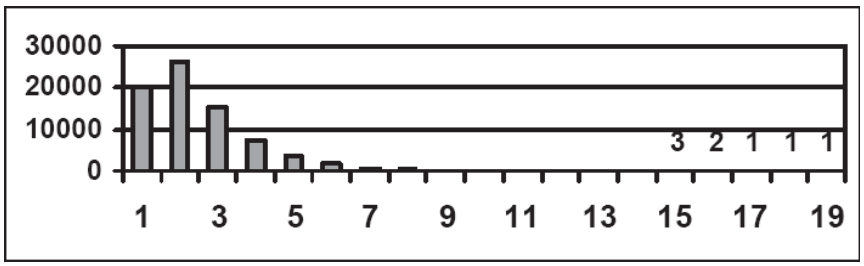


Figure 5.2 Distribution of the number of words in biomedical entity names. Data as found in the Genia V3.0 manually annotated corpus. Diagram from (Zhou 2004).

The challenge

Modest as these goals may seem, they are already a hard nut to crack. Usually a combination of dictionaries and manually devised rules is used to recognize the different appearances that terms can take (like plural, conjugation, and more).

The main difficulty is the lack of *standardization* of names. In biomedicine, a gene usually has many different synonyms, like CDKA, CDKA1, CDKA;1, cyclin-dependent kinase A, CDC2, Cell Division Control 2 and CDK2 all refer to the same gene or protein (for Arabidopsis genes, this can be inspected in the TAIR (The Arabidopsis Information Resource) gene name and synonym lists). This freedom in nomenclature gets even more complicated by the observation that in biomedical literature, often very long (multi-word) gene names are used. As figure 5.2 shows, entity names of 4, 5 or more words are quite common (Zhou 2004), take as an example the gene name 'Incomplete Root Hair Elongation'. This infers risks for unintended reorderings or conjugations of subterms, which causes more difficulty to design general entity recognition rules. See (Erhardt 2005), table 2, for a list of examples illustrating how one word-stem leads to many different variants.

For example it lists 14 variants for the term 'lipid' (lipids, lipides, lipidate, lipidated, lipidation, lipidizing, etc).
Furthermore, human language in general is complicated by *ambiguities,* in particular in biomedical nomenclature. First of all, common English words are sometimes also gene names, like hairy, cat, diabetes, who and how. And second, particular gene names or synonyms can refer to more than one gene (Erhardt 2005, Jensen 2006, Leser 2005). Additionally, publication-specific acronyms can interfere with official gene names (Erhardt 2005). All this shows that context analysis or 'knowledge of the world' is also a requirement for the assessment of the meaning of certain words.

Evaluation
Assessments of literature mining algorithms have been made, and notably by the BioCreAtIvE challenge (Critical Assessment of Information Extraction in Biology) (Hirshman 2005, Kinoshita 2005, Yeh 2005, Colosimo 2005). Evaluation of participants' entries showed that despite the obstacles, many entity recognition results were already useful for applications, with precision (correct proposal rate) in the 90%s, and recall (detection rate) up to 85%. At least four groups were able to achieve a weighted score of over 80%, which is notably still considerably worse than e.g. scores obtained for identifying persons and locations for online news (90-95%) (Blaschke & Yeh 2005). Evaluations of Leser et al. also indicated an 85% weighted score (Leser 2005). However, they noted the unreliability of these scores due to overfitting to a golden standard that includes too few training examples. Rzhetsky et al. also reported a lowering of scores when larger test sets were being used (Rzhetsky 2004), with results that rather ranged in the mid 70s and 80s for precision and recall.

Applications
An interesting application of Entity Recognition is the web-application iHOP (Hoffmann 2004, 2005). iHOP has scanned millions of PubMed abstracts (the summary of a publication) in a hunt for 80,000 biological molecules. It now allows visitors to search for a term, upon which it returns a list of sentences in any publication where that term appears. Hereby the term is highlighted, as well as other terms in the sentence, which are again clickable for a search. This website is a success story with hundreds of thousands of hits per month (Fernández 2007), which also stresses the need for biological information categorizing services. A drawback is still that the returned results form a long list of full natural-language sentences, which need to be re-interpreted again by every visitor. Clearly, information extraction is not that easy.
More applications are to be found in biomedical literature retrieval systems, like to some extent PubMed, and more advanced methods like Textpresso, which scans the worm *C. Elegans*' literature (Müller 2004).

5.2.2 Task 2: Information Extraction: formalizing the facts

The goal of Information Extraction is to extract predefined types of facts from biomedical texts, and in particular relationships between biological entities. To deduce meaning and intended relationships from a sentence, Natural Language Processing (NLP) techniques are most often applied (The more basic evaluation for co-occurrence is not discussed here).

First, Entity Recognition results are taken as building blocks. Then these terms are labelled with a part-of-speech tag, like 'noun' or 'verb'. Next, the syntax of each sentence is analysed; this means it is examined how the terms inter-relate, or the way how the different parts-of-speech come together in the sentence. Finally, a set of rules must be used to scan the syntax tree and determine one or more types of sought-after semantic relationships (Jensen 2006).

The challenges

Due to the inherent complexity that natural language poses to computer algorithms, automated Information Extraction currently only attempts to extract a limited number of information types from text. For example, one could direct its focus only to seemingly clear statements like "protein A phosphorylates B" (Yuan 2005), "protein C binds D," or "protein E activates gene F". However, as for example Rzhetsky et al. quickly realized, "even this 'easier' task is extremely difficult to perform correctly" (Rzhetsky 2004). One of the reasons is that natural language is frequently laden with ambiguities, for example, lexical ambiguity like "eating snakes *is* dangerous" versus "eating snakes *are* dangerous"; or syntactical ambiguity like "he hit the man with the bat" (does 'the man' or 'he' hold the bat?); or semantic ambiguity like "to grow tea" versus "to drink tea" (the former is the plant, the latter is the derived drink); see (Erhardt 2005).

The above ambiguities suggest the heavy reliance of our interpretation of text on context and on background knowledge that we learned from experience. Indeed, to truly understand language, we need a firm knowledge of the world, plus some reasoning capabilities. Take for example the sentence "The cat ate the fish, and now it's dead". Everyone with knowledge of how cats can cunningly stare at an aquarium would know that this back-referencing "it" refers to the now-dead fish. But suppose that the sentence was preceded by another sentence: "The fish was poisoned." Then our knowledge of digestive transfer of poison would allow us to do some reasoning and conclude that the "it" now refers to the cat that has died. This example illustrates the general principle of how natural language may be intertwined with many aspects of human knowledge and intelligence. This suggests that it may still take several decades before it is possible to let computers develop a complete reference framework with 'deeper' understanding of the surrounding world. This would, via disambiguating reasoning, constitute a definite impact on full language comprehension.

One further aspect of biomedical Information Retrieval that we want to mention is that many systems currently limit their scans to the abstract of publications only. While it may be dense in information, it may also be too compact to cover the extent of the paper's topic. For example (Colosimo 2005) assessed that abstracts contain only a fraction of gene names mentioned in the full text of a paper (25% for Fly and 36% for Mouse). Also (Corney 2004) found that typically less than half of the available information can be extracted from the abstract. Still, given the fact that abstracts are frequently the only freely available part of publications, this emphasis on abstracts is understandable. Luckily, the movement towards easy and open access to scientific literature is gaining ground, facilitated by for example PubMed Central (Wheeler 2008). But opportunities bring along problems too:

usually a publication's full text shows unbalanced information distribution. While the Abstract and Results sections may be rich in information, Materials & Methods may contain much detail that should not be considered as 'new results'. On the other hand, an Introduction section may rephrase results discovered in other research, and the Discussion may contain proposed, but not yet completely verified hypotheses.

As Rzhetsky et al. summarize: "The field of analysis of biological and medical texts is replete with exciting unsolved problems, problems more than sufficient to entertain myriad of researchers for many decades" (Rzhetsky 2004).

Applications and evaluations

Given the observation that Information Extraction, apparently an extremely complicated task, builds upon Entity Recognition, which is a non-trivial task either, it is understandable that expectations should not be put too high. Various text-mining efforts and services have been constructed by now, see also (Krallinger & Valencia 2005), of which we will address a number.

For example, (Yuan 2006) offers an online literature mining tool, specific for protein phosphorylation. It is supposed to extract phosphorylated proteins, phosphorylation sites and protein kinases from Medline abstracts (Hu 2005), and they claim "excellent performance." However, we have repeatedly tested the system, and witnessed how it performed poorly on 10 randomly selected abstracts. For example, for the top 10 articles from a PubMed search on "protein phosphorylation" on Jan 31, 2007, the evaluation score was only about 1 in 3.

Observations like this support Jensen et al.'s criticism of what they call "The jungle of quality estimates" (Jensen 2006). They indicate that using non-representative subsets of Medline can severely affect precision and recall estimates. Typically, a full text corpus will expose more than just the covered types of sentence-structures, which can make the predefined rules inapplicable. Also, it will introduce more 'noise' information in between the relevant information, making the assumption that 'there is' information less often valid.

Another system is BioRAT. This one exceptionally uses the full text of publications, from which it reportedly extracted considerably more information than from abstracts alone (Corney 2004). BioRAT achieved a little over 50% precision and mid-40% recall on full-length papers. A NLP-miner used for the Kinase Pathway Database (Koike 2003), reported reaching around 87% precision, but only 25% recall, illustrating a balancing trade-off between precision and recall. Donaldson et al. filled their molecular interaction database BIND with results coming from a text-mining harvest (Donaldson 2003). They stressed the necessity for human review of text-mined results, and pointed out that the combination of manual and automatic methods had lead to a large reduction in manual curation time. Some other initiatives of relational information extraction were carried out by (Chen 2008), (Feng 2007), (Krallinger & Padron 2005), (Rinaldi 2006, 2007) and (Tzong-Han Tsai 2007).

As part of the BioCreAtIvE challenge (Hirshman 2005, Blaschke & Leon 2005), participants were asked to automatically assign GO annotations to human proteins. Results for this especially complicated task showed a poor recall of 1 to 10%, again with a trade-off of higher recall for lower precision. They concluded that this demonstrates that "current systems are not yet able to produce satisfactory results for the extraction of biological information, especially where it requires complex extrapolation and integration."

5.2.3 Conclusion

Both the areas Entity Recognition and Information Extraction have made considerable progress in the past years. They do have the merit of scanning huge amounts of text and returning results that would otherwise rarely be found, but they only provide the tip of a proverbial iceberg. These automated techniques are only able to scratch the surface of the existing information wealth, and their results should still be manually reviewed since the approach is still very error-prone. Information extraction software is by far not ready to be used on its own, certainly not when the goal is the reliable, knowledgeable and complete fact extraction from biological literature.

5.3 Current manual text curation efforts

Several repositories constructed from manually curated annotations already exist, like GO (Gene Ontology Consortium 2000, 2006, 2008) or MIPS (Mewes 2007). But this mainly constitutes labelling genes with one or more functional category labels. In this section, we focus on initiatives that go one level deeper: efforts that deal with relations between bio-entities of any kind.

5.3.1 Manual annotation for text-miner training sets

A number of corpora of manually annotated publications have already been constructed. However, these are primarily intended for supporting the training and validation of automatic text mining techniques (Wilbur 2006). They usually do not envision the purpose of biological information collection on itself, in the sense of providing easier access to it for biologists. A consequence is that they generally don't support the full range of information-types that are present in a paper, although Kim et al. have recently made interesting progress in this direction (Kim 2008). Another consequence is that they usually require from the annotators that they painstakingly mark all the utilised terms in a sentence with a linguistic annotation, such as a part-of-speech tag. This will help text-mining methods to obtain an exact mapping between full text and translated information fragments; of course, that is if such an exact link is possible in the first place. But it does not motivate biologists to join such a curation effort on a larger scale, for the purpose of only the knowledge collection itself. Furthermore, these manual curation efforts, different in scale as they may be, in essence constitute rather isolated efforts of text-curation.

One of the largest manually annotated corpora is GENIA (Kim 2003). Version 3.0 consisted of 2000 annotated Medline abstracts, of which 1000 were recently re-annotated with enhanced semantics. GENIA also includes a taxonomy, a dictionary that assigns terms to a tree structure of categories, such as 'tissue' or 'protein subunit'. Another corpus is the protein name-tagged and protein modification-tagged iProLink resource (Hu 2004), which includes several hundreds of abstracts and full-text articles. Another example is GENETAG (Tanabe 2005), which is a tagged gene and protein name corpus of 20,000 sentences. 15'000 of these were used in the Entity Recognition part of the BioCreAtIvE competition. A final example is the much smaller corpus BioInfer, which is annotated, however, with extensive linguistic detail (Pyysalo 2007).

5.3.2 Manual curation for biological knowledge augmentation

Concerning manual text curation projects that do not solely serve text-mining training purposes, only a few undertakings have come to our attention. A first one is HyBrow (Racunas 2004, 2006). Via a web browser interface, it allows to enter various kinds of information in term-fields that have a fixed structure. Its main purpose, however, is the direct application of the collected information for computerized reasoning: based on an extensive set of manually entered and detailed rules, its goal is to create and test biological hypotheses via rule-based logic.

A second noteworthy initiative came from Kuhn et al. (Kuhn 2006). They are developing a general knowledge-capturing language called ACE (Attempto Controlled English), and in a case study they applied it to capture biological information. ACE is a rich subset of the English language that appears perfectly natural; although reminiscent of 'kindergarten' speech. But being a controlled subset of English it is in fact a formal language, meaning that computers are able to process its syntax. In the case study they explored in how far it could be applied to capturing the main aspects of protein interactions in the abstracts of biological publications. They took over 450 abstracts and succeeded to represent 56% of the protein interaction information completely in their language, and another 23% partially. Note that its similarity to natural language holds some danger to introducing ambiguity (see its online description on http://attempto.ifi.uzh.ch/site), but this is probably a natural trade-off against expression power. Kuhn et al. also argue that authors of new publications, next to their task of writing a standard abstract, could also formalize their findings into this or any structured and computer-readable format. Still, this message should be chimed much louder throughout the biological community, as well as a feasible infrastructure should be invented and built to make this happen.

5.4 Conclusion

Given the current state of technology, we came to the conclusion that neither single manual curation efforts, nor all-encompassing automated text-mining harvests can fully still our information hunger. Neither of these methods will be able to create both a large or highly reliable resource of biological information. This would leave the biological knowledge that has been produced so far, or that will be produced in the years to come,

basically not computer-processable or humanly overviewable. Therefore, in chapter 6, we lay out our bold proposition, which is the community-based manual curation of biological literature. Based on our insights from a first prototype we built, and from feedback we received from several directions, we will be able to describe many of the aspects that are required to make such a cooperated undertaking possible.

Chapter 6

MineMap: Extract, Visualise and Explore biological information

6.1 The solution: Community-based Manual Text Curation

6.1.1 The concept

The majority of experimentally verified biomedical information is currently available only as unstructured text in biomedical journals. In the previous chapter, we concluded that there is no clear-cut solution yet to make this information easily available to a researcher, or accessible to computational methods. Computer algorithms are capable of processing vast amounts of biomedical articles, but the information they extract is not of a satisfyingly high quality yet (Dickman 2003, Jensen 2006). On the other hand, existing but isolated manual curation efforts are able to extract relatively little, but more exact information; but they are not and will not be able to keep up with the ever-increasing amount of publications (Baumgartner 2007).

> Therefore, it appears that the inevitable solution is to engineer a widespread, organized effort of Community-based Manual Curation of biological literature, and this for a wide variety of information types.

6.1.2 The supportive infrastructure

To support such a community-based text curation project, we have designed a novel, multifaceted infrastructure: a system that combines new concepts and algorithms in an original way (see figure 6.1). We have subsequently programmed this new combination of various algorithms, forming a first prototype of this new software suite. This enabled us to put the system design to the test at several occasions, and to receive considerable feedback. As a result of these hands-on experiences, we were able to acquire new insights, and we have addressed some of the most critical issues that were spotted along the way. While it may still take considerable work to bring such a novel project to full fruition, we believe that we have laid a firm basis for a new direction in the biomedical sciences.

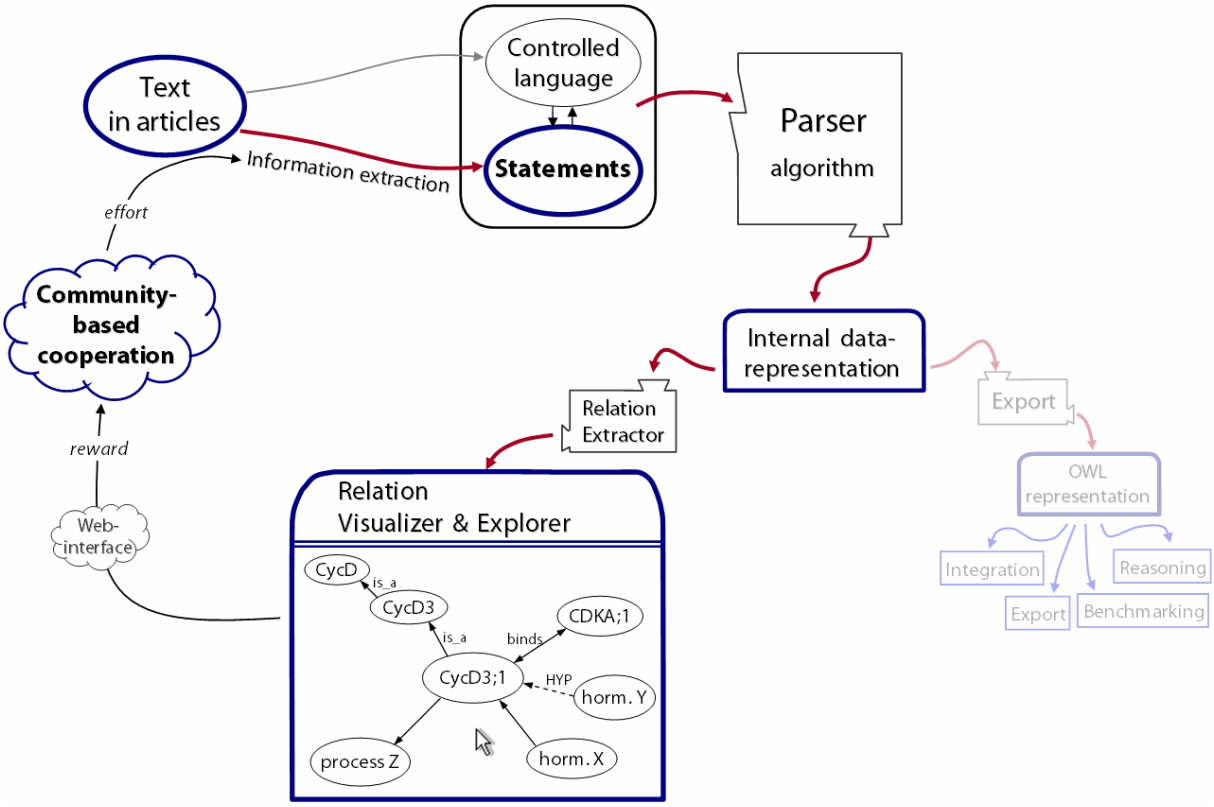


Figure 6.1: Outline of the MineMap project. The schema of the MineMap infrastructure and its various components it shown. We refer to the main text in this chapter for detailed explanations. The several representations of literature information discussed in this chapter are outlined in bold shapes. Some important data processing program-modules are shown as input-output process boxes. Data conversion flows are indicated as bold arrows. Possible alternate information output roads are drawn transparently. A cloud shape represents the internet.

Overview of the system's main aspects

During our research, we have identified several important aspects that we found to be required in an application for supporting community-based text curation. Several of these aspects are reflected in the modular design of the software, as illustrated in figure 6.1. Before we go into a detailed explanation in the following sections of this chapter, we commence with an introductory overview of the system's main aspects:

- **A controlled language Syntax** and its parser (i.e. the algorithms to interpret the statements in that language). This language should ideally be able to capture as much as possible the biological information as it is found in literature. This language has to satisfy the conditions of being both human manageable and computer interpretable. Therefore, it has been, and should continue to be, developed as a concise and structured format.

- **Common vocabulary and input assistance**. Since annotations coming from different annotators should be combinable, every annotator should use the same vocabulary. This makes it is vital to provide them with a hassle-free access to unifying and well-defined terminologies. In other words, a quick ontology lookup service must be implemented.

- **A rewarding Visualisation**. Annotators are not robots but human beings, who will sacrifice their time to annotate for their own and the general good. Therefore people

will need a reward for the effort. We provide this in the form of a visualiser that offers a pleasing representation and an interactive browsing experience on all the (potentially sizable) collected information. The visualiser module will represent each term ever used in an annotation, in the centre of a web of connected entities (proteins, complexes, organs, etc).

- **Web environment for cooperation**: We have recently entered the age of the cooperative, community-based internet projects, were many people add a small contribution to create a result of unseen magnitude. The logical next step is to take this paradigm and translate it to a scientific field like biology and biomedicine. We envision a web application that is somewhat reminiscent of a wiki (like Wikipedia), but with some major differences. First, the contents should now consist of structured information of which the meaning is computer-interpretable. And second, to compensate for putting an effort into this, biologists will expect a return on investment. They need to be able to access this information in a more pleasing and especially integrated fashion (see the Visualiser aspect, and figure 6.2). At this moment, we provide the prototype for this application as a Java applet connected via PHP-access to a central MySQL database. This makes our application accessible from anywhere on the internet.

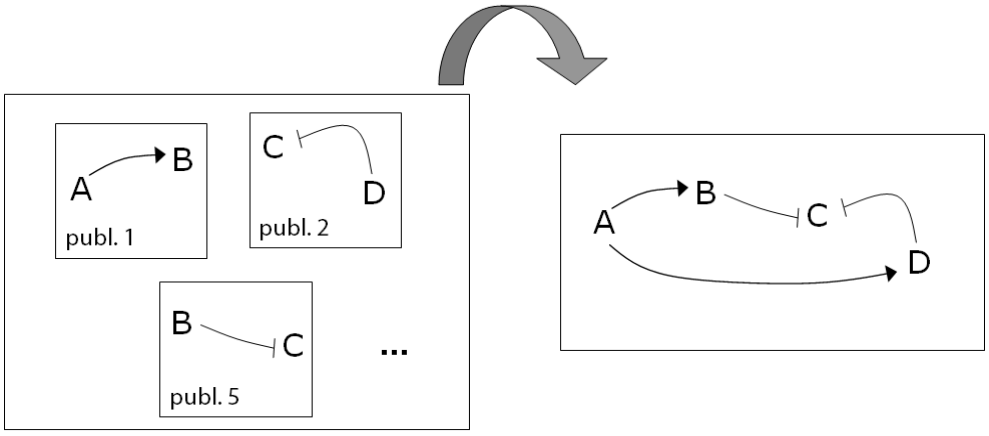


Figure 6.2: Integration of previously separate information. Many experimental results that were previously described in separate publications can be integrated into one combining overview. Our visualiser module takes care of this, but therefore it is essential that the published literature information is first transformed into a computer-readable format.

The name "MineMap"

We have named our software "MineMap", since it supports the process of mining and subsequent mapping of the collected biological information. Also, it is a pun on the term "mind-map" (with a 'd'). The visualiser represents the integrated information in a way that reminds people a lot of a mind map, which is a structure of related concepts around a core idea; although the display is also interactive and dynamically browsable.

Outlook & Applications

This framework for converting the vast quantity of findings reported in natural language into a concise, structured format has the potential to open several new doors for biomedicine. First, harnessing the power of computer automation on the cooperatively gathered information will make the researcher's quest for information easier. Secondly, an expectedly vast computer-readable resource of biological information offers the possibility

to spawn al kinds of new software tools to delve in this information (offering new representations, making integrated mash-ups with other sources, inferring implicit knowledge, etc.). Thirdly, a resource like this would be a goldmine for the text-mining community, making it possible to train the next generation of literature mining algorithms. And perhaps more boldly, as a structured bridge between computers and fully flexible natural language it could support significant advances in the intertwined fields of text comprehension, artificial intelligence and reasoning.

6.2 Aspect 1: The controlled language: Design principles

In this section, we lay out the design principles of our controlled language, a structured format with the capability of capturing much information from the molecular-biological literature. This will serve as founding material for section 6.3 which gives a concrete description of the basics of this format, and section 6.4 which goes into further detail.

6.2.1 A historic view on the structured format's origin

When we started the MineMap project, our objective was still modest. We explored the possibilities for a method that allows taking quick but structured notes when reading publications. If this would be feasible, then based on this structured information, a computer could subsequently combine all our extracted information and compose it in a convenient overview, an easily browsable visualisation.

This would mean a big leap forward, because when we, as mere human beings, read a publication in scientific literature, chances are that we have forgotten most of it within a week. We may be able to recall the general idea behind the article, and perhaps some of the most striking facts, but for most of the details we would have to return to the journal article and re-read the text. Alternatively, we may have prepared for this. Many of us take notes of the most interesting parts in the paper, as a short summary, and this can be used as a quick reference to find certain facts back. But when the number of read articles increases, our large stack of collected notes itself can easily transform into a body of text as opaque as the original full-text publications.

Therefore we aimed to develop a novel *controlled language* to capture information extracted from biomedical literature. It should be a concise, structured format that is easy to master by a biologist with minimal training in text curation. Yet, it should also be unambiguously interpretable by a computer program. This will enable software to combine all these human-curated textual notes and augment their accessibility. For instance they could be browsed as a web of connected entities (proteins, complexes, organs, etc), shared with fellow scientists, or integrated with other data sources.

6.2.2 Biological information variety

Covering the information plentitude

The information that a molecular biologist needs to extract from literature normally includes a wide variety of facts. It is typically not limited to a single biomolecular relation type or to the molecular interaction scale alone; often connections are made with higher-level observations as well. Therefore, our structured format was designed to capture as *broad a diversity* of information types as was possible for our first design. For example, it not only allows to express different kinds of interactions between biochemical compounds, it also foresees the declaration of links between protein abundances and cell growth or organ morphology, or the description of transgenic effects, localisation, cross-species observations and more. It can even be applied to describe parts of interaction diagrams, or partially supported but interesting hypotheses, and fuzzy expression or activity profiles. In summary, the direction of our concept of collecting and structuring varied information from literature, should be aimed to eventually support a field as general as Systems Biology.

A growing project

Although we already support the coverage of a broad informational diversity, we also realise that there are still various kinds of information or informational details that can not be covered in our language yet. This originates from the practical unattainability to find out all the requirements of the many biological niches, at the time of the first specification of this system. Still, thanks to all the feedback we received during the hands-on test sessions that we organized, we could already line out a considerable list of new ideas and feature requests that could be included in the next version of the language. This will be discussed at the end of this chapter. Note that it is technically more advantageous to adapt our parser program and all its dependencies only at a point were the language has reached a stable new level. So instead of adapting the parser many times, we have chosen to use our time for reprogramming other, more critical bottlenecks first. Furthermore, we have perceived that a complete, all-comprehensive language for biological information will be no easy job, but it does remain one that needs to be tackled in the near future. In any case, with our first design, we already realised a jump start for this structured format and for MineMap.

6.2.3 The literature basis for the first design round

Our perspective as network modellers

In our aim to develop a quite general language, we started from a number of publications on the topic of Cell Cycle control. We converted all the interesting information into some kind of structured format, whereby the interestingness was judged from our own perspective as dynamical modellers of biomolecular networks, especially the cell cycle (see the SIM-plex chapters). In spite of this initial niche, we could already include a plethora of information types like activation relations, phosphorylation, gene expression and protein activity profiles, transgenic experiments, functional classification and much more. Note that the cell cycle genetic network is extremely complex due to its iterative temporal dimension, its cyclical feedback loops and its many regulating control checkpoints. So a format able to capture all the information required for modelling this kind of network will certainly be well-suited for modelling genetic networks in general.

The emergence of a structured format

Given our plant research background, we took predominantly review publications of the model plant *Arabidopsis*. We chose to use reviews because of their high information and information type density, which was ideal to design our language. Note that one can argue that for future annotations, the original research papers should be considered too, as reviews could introduce an extra level of abstraction. While we were reading and extracting facts from these review articles, the emerging controlled language that we developed went through a number of design and redesign phases. It gained more and more consistency while being extended continuously, until a certain saturation level was reached where the available format was able to capture the majority of new information that was encountered. This level was reached after about 10 review papers, and it signified the end of the first development phase. We had composed a draft syntax and a comprehensive repertoire of relational symbols for our language, and we could now continue to design and program a parser for it. The publications we used were: (Beemster 2005, De Veylder 2001, De Veylder 2003, Inzé 2005, Menges & Murray 2002, Menges & Hennig 2002, Mironov 1999, Nurse 2002, Vandepoele 2002, Stals 2001).

Resulting scope of the language

Our goal was to capture an information range as broad as possible. For several types of information, especially on the molecular interaction scale, the test sessions showed that our language turned out to be satisfyingly powerful. For some other information types that fell less in our initial area of attention and for which we could only foresee a basic initial coverage we concluded that significant development would still be required to promptly satisfy all needs. This was especially the case for transgenic descriptions, where a detailed identification of the many complex ways to construct a transgenic line can be desired.

6.2.4 Syntax and relationship semantics

Language = vocabulary + syntax

Before we go any further in describing this structured format, or language, we have to make a note of what *a language* actually is. There are two pillars that make up a language. The first is a *vocabulary*: this is basically a dictionary of words and their meaning (the *semantics*). The second is a *syntax*: this is the structure of how the words come together and form groups like clauses or sentences.

Application to the MineMap language

In our structured language, note that the *words* that describe *relations* are in fact *symbols*. For example, "A activates B at the G1 phase" is written as "A -> B @ G1 " (as will be extensively explained later on), and so the words for the relation types "activates" and "at" are written as the symbols "->" and "@". This is an important observation. It means that the definition of our language consists of the following two parts. First there is the vocabulary of the relation types, represented as symbols, plus a not-inherently defined free vocabulary of bioentities (this will be further restricted in a later section that describes ontologies). And second there is the syntactical definition of our language, which defines how the terms describing bioentities come together with the relational symbols and form meaningful statements. As a remark, one may suggest to use words for the relationship

symbols as well (as a future development). However, biologists are already used to represent relations between genes, proteins, processes, etc with graphical symbols in diagrams, so extending this habit to our textual format gives the language a jump start in usability.

6.2.5 Notation: inspired by the graphical notations

Graphical notations are widely used to transmit knowledge about biomolecular interactions in an intuitive manner. Many publications mainly use the classical activation and inhibition arrows (--> and --|) to represent molecular interactions. Still, this is often incomplete and ambiguous with respect to the interaction mechanics (Pirson 2000), for example an arrow can be used to represent indirect activation, direct transcriptional activation, as well as protein transformations. A number of graphical notations have been designed to address some of these issues, such as Kohn's Molecular Interaction Maps (Kohn 1999, 2006), and Kitano's powerful graphical notation that even allows the definition of many phosphorylation sites and their protein activating properties (Kitano 2003).

In order to align with the expertise in this field, our textual representation was designed to correspond to several of these intuitive graphical notations, at least for molecular interaction symbols. To give a preview of what the language's statements look like, we present a number of molecular interaction examples that were based on the Kohn and on the Kitano notation. Figure 6.3 shows some typical interaction symbols and a small network built with such arrows.

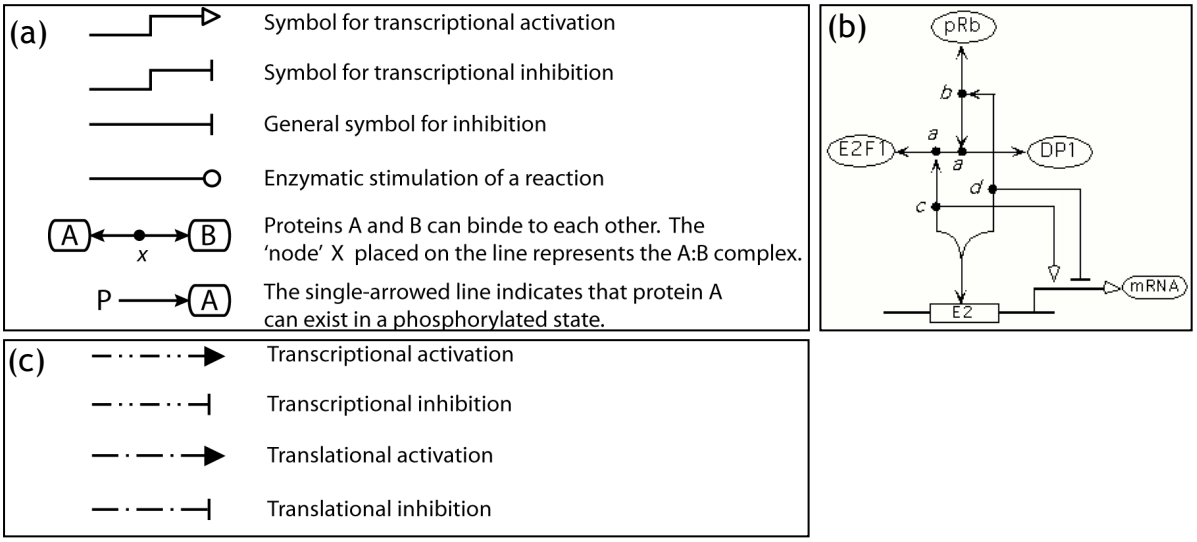


Figure 6.3: Graphical notations as a basis for the MineMap stenographic notation language. (a) Some typical Kohn interaction arrows and symbols. (b) An example molecular network built with such symbols (c) Just a few of the many Kitano notation interaction arrows.

Table 6.1 shows some example translations from Kohn's and Kitano's graphical notations in figure 6.3 into our textual notation. Most examples are based on the Kohn notation, because this notation was already sufficient to cover a large part of the molecular interaction types described in the reviews that we used to develop the language.

As literature is produced and read by a large and international scientific community, we attached much importance to an essential design principle for the controlled language: all symbols should be found on a standard keyboard. As many scientists should be able to use this method, there should be no limitation by the type of keyboard they use. This limits the vocabulary's design to use only the basic Latin letters, numbers and punctuation marks found on all types of Qwerty or Azerty keyboards.

Textual symbols	Meaning
A -> B	activation
A -| B	inhibition
A -s> B	transcriptional activation ("-s>" symbols the step-up arrow)
A –s| B	transcriptional inhibition
A <-> B	binding
… -> (P -> B)	phosphorylation of B (shorthand notation)
… -> (B -t> B[P])	phosphorylation of B (with a transformation-arrow)
A -.> B	translational activation
A -..> B	transcriptional activation (Kitano's alternative to "-s>")

Table 6.1: Some example shorthand textual notations and their meaning. All examples are inspired by Kohn's and Kitano's graphical notations.

6.2.6 Notation: an example

To give a preview of our notation method's power, we translate a piece of information that could easily appear in any molecular biological publication: see figure 6.4.

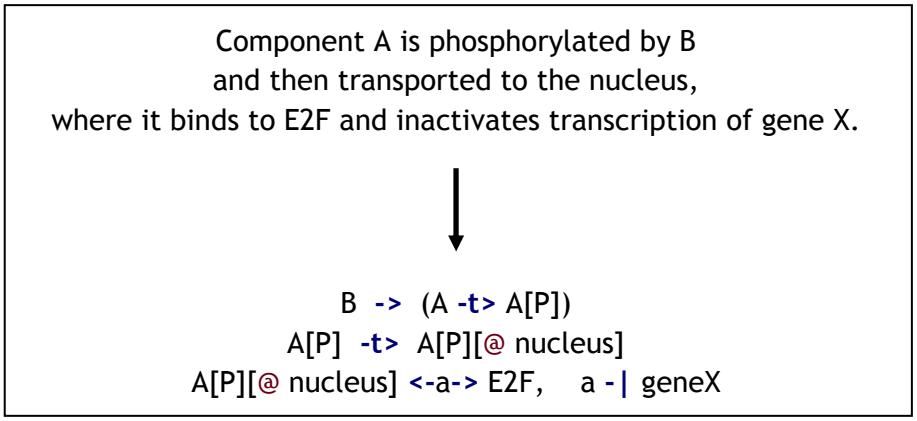


Figure 6.4: Introductory example of some natural-language information commonly found in biological literature, and its translation to structured statements. See the main text for a more verbose description of the translation.

In figure 6.4, the example sentence is translated as three separate statements. The first one says that *B* stimulates the phosphorylation of *A*. The "->" arrow represents activation, and the "-t>" arrow represents any kind of transformation (which the distinguishing "t" of "transformation"). The entity "A[P]" stands for a modified version of the A, namely one that is modified by a protein phosphorylation (which is often indicated via an appendix "P" in protein interaction diagrams). So in our language, modifications are indicated by adding something in square brackets behind the original thing. As an aside, this line could also have been written via the Kohn-notation shorthand "B -> (P->A)".
The second line says that the phosphorylated form of A is transformed to another form of A, which is still phosphorylated, but now also located in the nucleus (so the "-t>" symbol is reused). Location can be indicated via the at-symbol "@".
The third line describes the phosphorylated form of A in the nucleus, which binds to E2F ("<->" means "binds"). The bound complex is temporarily named "a" (written via "<-a->", which is taken over from Kohn maps), and this "a" is declared to inhibit geneX.

In summary, the statements literally say:

line 1: B stimulates: the transformation of A to phosphorylated A.
line 2: Phosphorylated A is transformed to a new type of A:
phosphorylated, but now also specifically located in the nucleus.
line 3: The phosph. A in the nucleus binds to E2F, and the bound complex
(which was temporarily named "a") inhibits geneX.

Though many more information types than this can be represented in our language, this forms a first hands-on introduction to where the in-depth description will lead us.

6.2.7 Human usability in the human/machine interface

The structured format had to satisfy the conditions of being both human-manageable and computer-interpretable, which puts it on the pioneering intersection between these two worlds. Research in this area has been limited so far. To quote Kitano: "Although there has been significant progress in machine-readable representation of networks, as exemplified by the Systems Biology Mark-up Language (SBML), issues in human-readable representation have been largely ignored" (Kitano 2005).

The three human aspects for a usable language
Three human aspects have to be considered in the design of a written language, which are learnability, readability and writability.

- For *learnability*, the symbolic notation (like the arrow symbols) makes a direct link from existing diagrammatic notation and the textual notation, and mostly offers intuitive shorthands. More symbols are described in what follows, but they all follow the same spirit.
- For *writability*, thanks to the symbolic notation, we have a concise, shorthand language that makes it quick to write, like stenography. Moreover, the dictionary lookup service with term autocompletion for the bioentities (see further on), further increases MineMap's writability substantially.

We also think that this assisted textual notation forms a speedier way to take notes from articles than it would be via clicking several fields and graphical buttons before a piece of information can be entered or edited, like in HyBrow (Racunas 2004). In MineMap, a publication's annotation is freely edited in a text pane, because one of our system's goals is to enable information collection without much interference. Note that, if desired, it would still be possible to let part of the input happen via a graphical user interface layer on top that could write to an underlying textual representation.

- For *readability*, we found that in this notation the symbolic interaction stands out. This makes the language quicker to read and easier overview than summaries in natural language.

As an illustration what a difference some focus on human-usability can make, we show a comparison between an information snippet in the MineMap format, and the same piece in a format like the Web Ontology Language (OWL).
The fragment says: "In fission yeast, phosphorylated Cdc2 stimulates Xyz":

- MineMap syntax (human readable):

  fission_yeast: Cdc2[P] -> Xyz

- Manchester OWL syntax (not designed for human readability):

  at_organism SOME fission_yeast
  AND
  activator ONLY ((phosphorylation EXACTLY 1) AND Cdc_2)
  AND
  activated ONLY Xyz

Required effort
Based on our experiences, we estimate that a person having some experience with the symbolic language can read and annotate a paper in at most twice the time it takes to just read it. Some parts of the text will typically not be considered (cf. the discussion about information density in chapter 5); some parts will be translated easily; and for some other parts it can take quite some time to disambiguate and formalise what the authors actually meant.

6.3 Aspect 1: The controlled language: Specification

In this section we move into more technical detail, and specify the variety of statements provided by MineMap. Most often, a *statement* is a representation of one single piece of information as captured from biological literature. For this description, it is impractical to fully report all the rules of possible combinations to form clauses and statements, in the way that the parser software is programmed. Instead, for user-friendliness, we will divide the general setup in topics, and for each topic give a few illustrative examples.

While most statements just represent one piece of information, like 'A -> B', for 'A stimulates B', two other constructs do not capture information. These are *mode definitions* or *comments*, which will be described before all other statements.

6.3.1 Technical facilities

Before using the language, it is good to know that one can insert a comment at any place. Comments are text that will be ignored by the parser program, and can be used to write down some reflections by the annotator. Note that this reflects how part of our inspiration also came from computer programming languages.

```
A -> B   //… or: How I Learned To Stop Worrying and Love Writing the Thesis.
/* One is free to say what one wants
   in a multi-line comment. */
a_statement_that_spans_more  ->  _
   than_one_line        //Split single lines with a space+underscore.
```

6.3.2 Mode definitions

Mode definitions are statements that work on the meta-level; they attach their meta-information to all the statements that follow. For example, they can tell what species (organism) the current publication is describing, or what subject (e.g. section title in the article) is covered by the following statements.

```
SPECIES: Arabidopsis
SETTING: sucrose_starvation              //The experimental setting.
SUBJECT: E2F role in G1/S transition     //(This can be free text).
```

Note: the person who extracted the statements from a publication is also required meta-information, as well as a reference to the original publication. However, this shouldn't be defined as a statement; instead the MineMap web-interface will keep track of this information based on the user's login ID and the selected article's ID.

6.3.3 Entities

Entities are the words that build information-containing statements. In most computer languages and also in MineMap, a space is used to separate these entities. Therefore, if a term consists of multiple words, it should be separated by underscores ('_') instead of spaces. (In fact, this makes the parser program considerably easier).

```
leaf_development
CDKA;1
```

One can also combine separate bioentities via the 'dot-notation', to further specify an attribute of the first entity. For example, one can talk about the expression of the gene CycX: "CycX.expr". This dot should be read as the possessive form, so "CycX 's expression". Along the same line, this notation can also be used with a few language-specific shorthand attributes, like the "expr" for expression, "prot" for protein, "RNA", or "DNA".

```
CycX.expr
yeast.cdc2
time_interval.begin
```

A note concerning the gene vs. protein distinction: in several species (like Arabidopsis, but not human), a gene carries the same name as its derived protein. In that case, the bioentity name in MineMap will represent both at the same time, and usually the context will specify which one it is (e.g. only proteins get phosphorylated). Notice that in many cases this is not even clear from the publication, as even human annotators disagree in 23% of the cases (Tanabe 2005). If it is necessary to explicitly distinguish between the two, then one can use:

```
gene.prot
gene.DNA
```

Square brackets are used to define a derived entity from the basic one. The examples below represent: "phosphorylated Cdc25", "protein A phosphorylated at the site T14", and a double phosphorylated protein:

```
Cdc25[P]
A[P,T14]
A[P,T14][P,Y15]
```

One can attach a small, free-text note to an entity, between curly brackets:

```
Cdc25[P]{active form}
```

It should be noted that entities are always assumed to possibly be a set. For example when saying that CycD3 activates something, it means that every member of the *set of* CycD3s activates it. So when someone (later or earlier) defines that the entity CycD3 is actually a set, by saying that 'CycD3;1 is_a CycD3' and 'CycD3;2 is_a CycD3' etc., all these activates-relations would also hold for the members of that set.
One can also explicitly define a set, which is usually used together with the "="-operator:

```
(CycA, CycB, CycC)
```

One can declare all kinds of set combinations. With a little imagination, one easily sees that "u" stands for union, and "n" for intersection in the examples below. Note that our language should had to be both shorthand and typable on most keyboards. For the set-difference operator, we use the backslash symbol (the forward slash is used for mathematical division, see later on).

```
leaf \ leaf_stoma
(A u B)  n  (C \ (D, E) )
```

One can add a unit entity after a number, for example:

```
duration = 5 h
```

Some support for quantities is also present, like "# A", to be read as "number of A-s", and meaning the number of elements in the set A. It is in fact shorthand for a special attribute: "A.number_of". Also, some fuzzy quantities are predefined, like "high" and "low", which can for instance describe qualitative protein activity profiles coming from Western blots. Note that it is generally not possible to attach values to this fuzziness; they are only meant to describe relative changes.

```
# yeast.cyclins   //(shorthand for the attribute 'number_of').
```

```
A = high
B = medium
C = low
```

6.3.4 Relations

The most basic relation, also used in many other information repositories (like ontologies), is the elementary parent-child relation, or "is_a" relation. For example, one can say that "CDKA;1 is a type of CDK". In MineMap this is written as "CDKA;1 (= CDK", with the mathematical set-inclusion as the relational symbol, reading out as "is a" or "subset of".
Note that both CDK and CDKA;1 should be thought of as representing *set*s here (with the latter as a singleton). Consider that it may be known that CDK comprises a number of different CDKA genes, CDKB genes, etc, and that there exists only one CDKA;1. But possibly in the future biologists could discover that there is again more than one type of CDKA;1. In any case, this is merely a conceptual matter.

```
CycD3;1  (=  CycD3
(CycA, CycB, CycC)  (= Cyclins
```

All the basic relational symbols are provided (equals, does not equal, larger than, etc) :

```
CycX.expr = high
 A  !=  B    //This is the common programming language operator "not equals".
 A  >   B
 A  <=  B
```

Homology between genes and proteins, or general similarity (a distinction can be made based on the context, the type of both entities) :

```
mouse.protA  =h  rabbit.protB
```

The most common activation relations are also available in the language. Note again that the "s" in the "-s>" operator is inspired by the step-up arrow as drawn in Kohn diagrams. As shorthand, the set notation can be used for each of the entities. For example "(A, B) -> C" would stand for "both A and B stimulate C". This statement is split into two separate pieces of information by the MineMap parser.

```
A  ->   B       //Activation stimulation (molecular interaction level unspecified).
(A, B)  ->  C   //Shorthand for: both A and B stimulate C.
A  -s>  B       //Transcriptional activation (alike Kohn's notation).
A  -.>  B       //Translational activation (alike Kitano's notation).
```

The inhibition relations are typed with almost the same symbols as the activations, except for the vertical bar symbol at the end '|' (also named *pipe*). Note that on most keyboards, the vertical bar symbol is depicted as a broken vertical bar '¦', to distinguish it from the 'I' (uppercase 'i') character. But when typed, it will likely appear as an un-broken '|'. The key is usually located next to the 'Enter' key, or on the '1' key.

```
A  -s|  B
```

The "~>" operator stands for "controls" or "mediates" and should be used when an influencing interaction is declared, but it was not defined whether this is an activation or inhibition. The perhaps less often used operator "-o" (alike Kohn's notation) stands for "enzymatically promotes a transition", so in "A –o B", the B should not be a biomolecular process, for example a transformation.

```
A  ~>  B            //General
A  -o   (B -t> C)   //(Notation like Kohn).
```

For transformations, one can use the "-t>" arrow. Although commonly drawn in interaction diagrams with the same plain arrow as for activation, we have to take away this ambiguity. For example in "A->B", A is the activator, while in "A -t> B", A is transformed.

```
A -t> B                 //Biochemical transformation from A to B.
A -t> A[P]              //Phosphorylation of A.
A + B -t> C
A + B -t> C + D + E
```

We still mention some special shorthands:

```
Abc -> (P  -> A)        //Abc stimulates the phosphorylation of A.
Abc -> (A -t> X)        //Abc stimulates the destruction of molecule A.
```

Finally, "<->" declares the physical binding of two molecules, as used in Kohn maps. As mentioned before, by placing a letter in the middle ("<-z->") one can subsequently tell something more about the bound complex, all in the same statement.

```
A <-> B
A <-a-> B ,   a -| C    //A and B bind, and the resulting complex inhibits C/
```

6.3.5 Quantities

It is also possible to perform some mathematics with entities:

```
(duration1 + duration2) / 2   >  5 h
cell_cycle.length  -  G1.length
cell_division_rate * duration
```

The "++" and "--" operators provide some convenient shorthand: "A++" is an identical alternative for "A = increased", and "B--" means "B = decreased". Note that "Increased" and "decreased" are both terms included in the PATO (phenotypic qualities) ontology.

```
cell_growth ++
cell_division --
```

6.3.6 Time and space constraints

The at-operator "@" is used to specify both temporal and spatial constraints. Whether it is space or time, can be deduced from the entity that follows the "@" symbol. On a historical note: in the original language specification, we provided both the "@T" and the "@L"

operators (for time versus space). Not only provided this unnecessary overhead, we also noticed that the "@"-notation was becoming useful to represent more than only time and space, but could be used for constraining-conditions in general. Therefore we dropped the T/L-appendix.

```
A <-> B   @nucleus
A -s> B   @S_phase     //Postfix-notation.
@S_phase:   A -s> B    //Prefix-notation.

A -s> B   @virus_infection      //Non-space/time constraint.
```

As a special provision for the set combinations, we also allow terms for the universal spatial and temporal sets:

```
Abc.expr=high  @always      //= ... is true "at all times".
... @(all\Golgi)            //= ... everywhere except in the Golgi app.
```

One can also use the space/time operator in the modifier part of an entity:

```
X[@t1]  >  X[@t2]           //"X at time t1 is larger than at time t2".
```

Finally, when we take this manner of writing and we reuse the transformation symbol "-t>", then we can define transportation, without inventing an extra operator. The following statement defines the transportation of Abc to the nucleus (literally, it would read out as: the transformation of Abc, to the Abc modified as being in the nucleus). Note that here, Abc's original location is not specified. However, this is often also not explicitly told in literature.

```
Abc  -t>  Abc[@nucleus]
```

6.3.7 Prefixes

One can specify, for a single statement, that it is valid only under certain special conditions. For example, to override the currently declared species (via a mode-definition)

```
yeast:  A -> sugar_intake          //Overrides the currently declared species.
mouse, frog, chimp:  A -> B        //Valid in all those species.
species(yeast):  A -> B            //Alternate notation.
setting(drought_stress):  A -> B   //Declares a special experimental setting.
```

Sometimes one may wish to enter an assertion that is only hypothesized in the publication. For this, one can use the specifier "HYP:" in front of the statement. While the information should be based on some leads, for now it is inconclusive.

```
HYP:  A -> B  @nucleus //The authors hypothesize that A activates B in the nucleus.
```

6.3.8 Quantifiers and logic

This section enters into the more experimental region of statements. First of all, we noticed that in literature, authors sometimes make general assertions like "This proves that there must be a protein that interacts with Abcd and that stimulates the G2/M-transition",

or "Most of the CyclinZ-s interact with Abcd." To capture these, we included quantifiers (exists / for all) and logic in our language. For example the first statement would be written as:

```
è protein :      <-> Abcd    &    -> G2_M_transition
```

This, with the "è" operator (or "é") as the mathematical existence operator, looks a lot like a mathematical formula. As that is usually not too user-friendly, we made a first step in the user's direction, and allowed the omission of the 'quantified variable'. Concretely, here this means that one isn't obliged to write "è protein: protein <-> Abcd". So the slightly clearer "è protein: <-> Abcd" would read as "there exists a protein *that* binds to Abcd".
As a remark: although the current vocabulary accepts only accented é and è for the exists and the for-all quantifiers, this would best be replaced by a plain 'e' and 'a' in the future, because of keyboard generality and portability considerations.
Furthermore, like in mathematics, one can use logical operators in the tail of these statements. Possible logical operators are: & (and), | (or, the vertical pipe symbol again), ! (not, as in "!=" for inequality), "=>" (implies).
Some more examples:

```
è CDKB :  CDKB <-> CycD4;1  & CDKB  -> G2_M_transition
è CDKB :        <-> CycD4;1  &        -> G2_M_transition
à CycD  :  è CDK  :  CycD <-> CDK
```

An assertion like "Many cyclinD proteins bind to a CDKB protein" can not be captured easily in commonly known mathematical terms. It would be too weak to use the plain existence-operator "è", since we know that there exist *many* CycDs. Therefore we provided (experimental) "fuzzy quantifiers": one can take a quantifier-operator and append a modifier to it, like in:

```
è[many] CycD :   <-> CDKB
```

Although to our knowledge, information like this can not be exported to any other format yet, still we believe it provides for an interesting idea.

6.3.9 Various other statements

We provide some basic support for defining transgenic phenotypes. The first example below says: "In an Arabidopsis CDKB1;1 overexpression line, cell division was decreased, but CycX's expression was high during the G1 phase". These statements don't provide direct molecular interaction evidence, but their indirectly implied clues can be useful for hypotheses about network structure, and for validating dynamical model simulations.

```
Arab[CDKB1;1++] : cell_division --,  CycX.expr=high @G1
Mouse[A++, B--] :  event,  thing --, property ++,  prop2 =,  prop3 = equal
```

Further support for dynamical simulation comes in the form of experimentally measured time courses: gene expression profiles or protein activity profiles over time. These are virtually always given as fuzzy descriptions (often only visually), as is reflected in the capturing statement given below. Note that this already lies on the border line between *information* extracted from the paper, and raw, uninterpreted *data*.

```
Abc.expr = [G1.begin: low,  G1: ++,  S.begin -1h: medium,  S.begin +1h: high]
Abc.expr = [offset=G1.begin,  0h: low,  0h-8h: ++,  8h-12h: =,  12-22h: --]
```

A last operator allows negating any assertion, except for the special transgenics or time-course statements. The following says: "It is known that A does not activate B in any way".

```
! ( A -> B )
```

6.3.10 Review: Some basic reference examples

Mode-definitions

SPECIES: Arabidopsis
SUBJECT: G1-entry

Language basics

```
(A,B,C)  (= ABC          //Symbol  (=  means  "subset of", "is a"
      A  (= B
CycD. expr = high        //Symbol  .   means  "attribute".
A  -t>  A[P]             //Symbol  [ ]  means  "modifier".
```

Relations

```
A  =  B                  //A equals B.
A  =h  B                 //A is homologous to B.
# CDKB > 2               //The number of CDKBs is higher than 2.
```

@-specifications, sets

```
A  ->  B                 //A stimulates B.
A  ->  B  @ G1           //A stimulates B, during G1.
cell_size++  @ leaf \ leaf_stoma   //Cell size increases in the entire leaf except in the stomata.
```

Prefixes

```
HYP:    A -> B           //It is a hypothesis that …
human: yeast.geneX -> geneY   //In human, the (inserted) yeast geneX stimulates (human) geneY.
```

Transformation

```
B  -t>  C + D            //B is transformed (splitted) into C and D.
A  -t>  A[P,T14]         //A becomes phosphorylated on site T14.
```

Quantifiers (exists / for all) & logic

```
è CDKB :  <-> CycD4;1  &  ->  G2_M_transition   //Some CDKB binds CycD4;1 & stimulates G2/M.
è[many] CycD :  <-> CDKB                        //There are many CycDs that bind CDKB.
```

Transgenics & time courses

```
Arab [CDKB1;1++] :  cell_division --,  CycX.expr=high @G1   //In a CDKB1;1 overexpression line, …
CycX.expr = [G1.begin: low,  G1: ++,  S.begin-1h: high,  G2: --]   //A CycX expression time series.
```

6.4 Aspect 1: The controlled language: Parser algorithm

The language we described in section 6.3 goes together with a strict syntactical definition. This rigorous list of rules was used to construct the parser algorithm. This is a Java program, for which we applied the JavaCC parser generator (like we did for the SIM-plex parser). It allowed to list our syntactical definitions, and to populate it with Java code that is executed whenever a certain statement type or statement fragment is encountered.

The result of the parser code running on a correct list of statements is the creation of an internal data representation called a *syntax tree*. For example, for the single statement

A[P] -> B @ G1.begin

the syntax tree would look like figure 6.5. It is composed of a number of nodes (like Control or Id), which are all Java classes of various types. This class-type stands for the node's semantics. In the figure, the node Control (representing our single statement) has two children of type Id: one for the activator "A[P]", and one for being-activated "B". Moreover, it has an extra specification node Time, for the "@ G1.begin" tail. Because "G1.begin" is composed of two fragments separated by the dot, its Id node has two IdFrag child nodes, one representing each part. For A[P] and B, there is only one IdFrag. But A[P] also has a modifier "P", which is reflected in the extra IdModStr node (see below). This syntax tree represents a semantic analysis, a partitioning of the flexible statements that were given as full text.

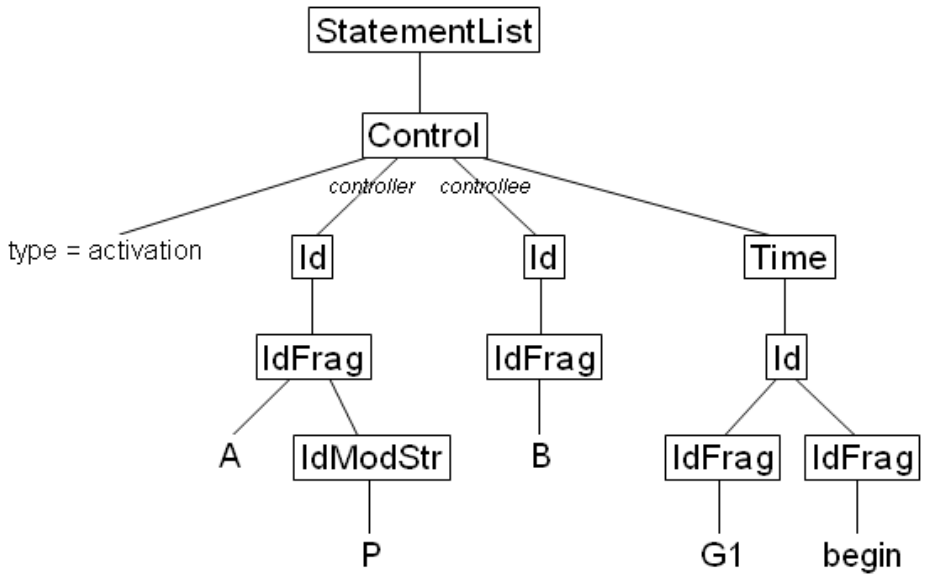


Figure 6.5: An example syntax tree. This represents the parser's output for the statement "A[P] -> B @ G1.begin". See main text for details.

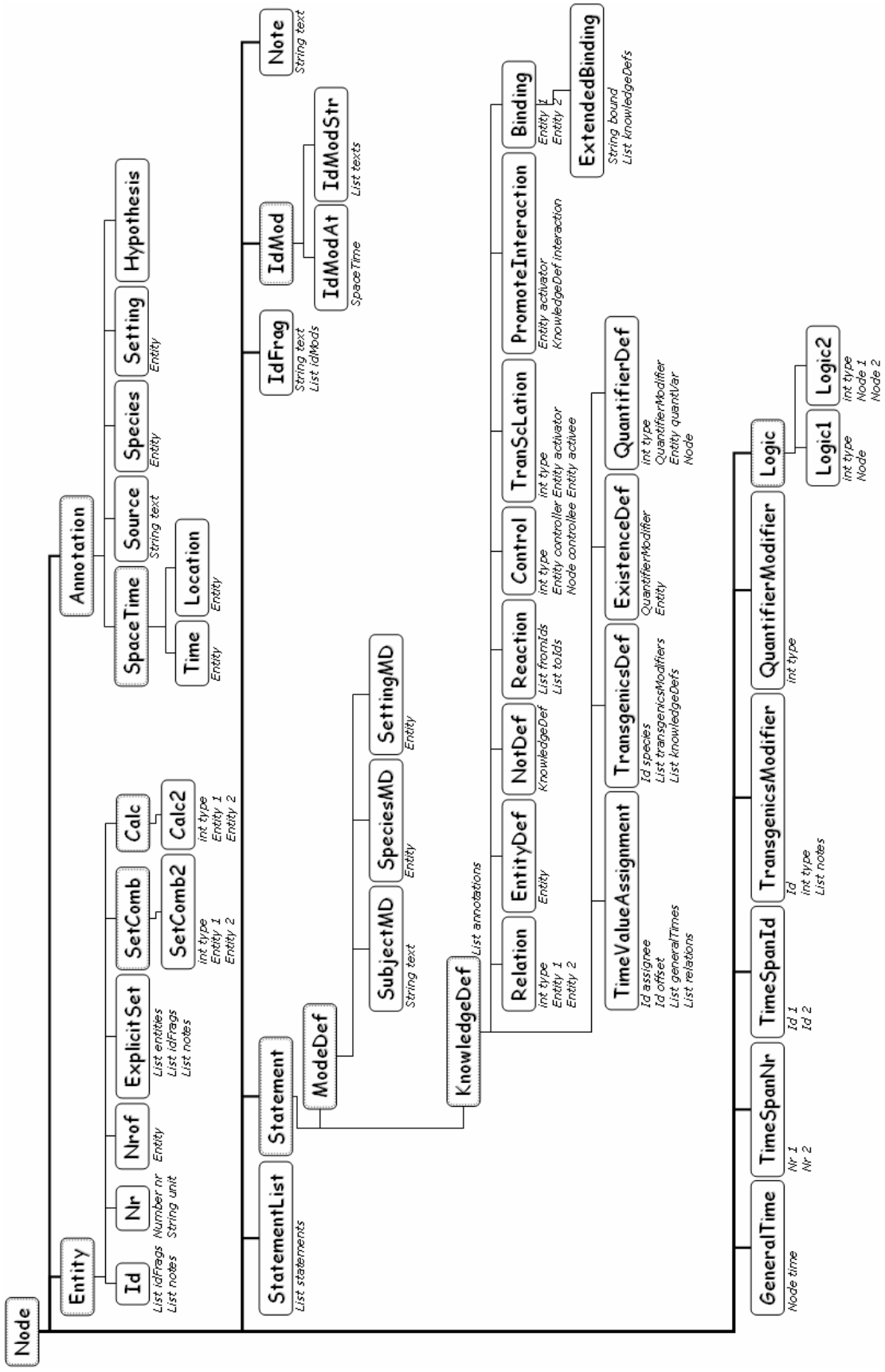


Figure 6.6: Class hierarchy of node types used by the parser. The parser converts a statement list of manually curated information into a syntax tree (not shown here), which is usually built from nodes of various types. The conceptual parent/child-relations between these types of nodes, or the Java class hierarchy, are shown here.

There are in total 51 types of such nodes in the MineMap internals. All these nodes are Java classes, which can be classified in a logical hierarchy: see figure 6.6. For example the node Control, which was used to represent our single statement, is a KnowledgeDef node, which is in turn a Statement node, which is like every other node, a descendant of the super-class called Node. A detailed, technical description of all of these classes and their possible connections with each other is well beyond the scope of this thesis; but it can be inferred from the language specification in section 6.3, in combination with figure 6.6.

6.5 Aspect 2: Common vocabulary, dictionary support

It is vital that all annotators speak the same language when their manually curated information is integrated into one overview. The syntactical part of this structured language has been covered in the previous sections. But as we already mentioned in section 6.2.4, we also have to pay attention to the other part: the vocabulary. We need a reference dictionary so that all annotators can use the same term for the same intended meaning. For example in Arabidopsis, one annotator may use the term "silique", while another one may use "siliqua", and yet another one may use the synonym "fruit".

Ontologies and other sources

A large part of the solution can already be found in *ontologies*. For our purpose, one can see an ontology as a set of terms, hierarchically organized in a tree structure via "is-a" relations. Each term represents one meaning, and for each meaning there is exactly one (preferred) term. A term can have a list of known synonyms too (for reference), but one should always use this preferred term. Each ontology intends to eventually fully cover a specific knowledge domain. Some examples of ontologies are the Gene Ontology (GO) (Ashburner 2000, Harris 2004), the Plant Ontology (PO) (Avraham 2008, Ilic 2007, Jaiswal 2005), and the Phenotypic Quality ontology (PATO).
Next to the ontologies that describe specific biomedical domains like molecular function, anatomy, diseases, processes, conditions and more, there are also gene names. Unfortunately, there is often no full standardization yet for gene names, and one has to work with more fragmentary gene name lists.
It should be noted that all these initiatives are a continual work-in-progress, and that some terms required for an annotation may not be available from any source yet; so MineMap does not constrain the vocabulary to only dictionary terms.

Dictionary lookup assistance

Often a single ontology will count many thousands of different terms. Also the gene list of a single organism will typically rise in the tens of thousands. Therefore, we need to give the annotators some user-friendly term-lookup assistance. We were inspired by the Ontology Lookup Service (OLS), a web-based tool that looks up terms across tens of ontologies (Côté 2006). The beauty of this service is that the lookup happens via the Ajax web-technology (lookup requests without page reloading), so even partially typed terms are already looked up and matched, while the user is typing.
We translated this concept into our own platform, and made it possible to include terms from ontologies as well as gene lists in our dictionary. When MineMap's users are typing in

the annotation text-pane, the lookup service automatically shows a list of suggestions underneath the word that is being typed. The term can also be automatically completed; replacing it by its main term in case it would be a synonym. Note that for this task, we wrote a novel Java module, as existing term suggestion and autocomplete modules were based on single-word textfields only (instead of a full text-pane), and none were offering asynchronous lookup services.

6.6 Aspect 3: Effort & reward: the dynamical visualisation

6.6.1 Visualisation

User-friendly access to the collected information will form a pillar for the success of a large-scale manual curation effort. The attractiveness of the information presented back to the annotator, is likely to be closely related to his/her willingness to continue in text-curation.

The optimal way to present this information, we believe, is in a way that most closely correlates to how we would envision the information ourselves, in our minds. When we consider an object or a concept, our thoughts immediately activate many concepts and associations around it (Motter 2002). If drawn on paper, this would look much like a *mind map*, which is a structured representation of a core idea with all related concepts around it. Furthermore, in our minds, we can hop from concept to concept when we look for information, patterns, or new connections. So realizing this idea in an interactive and attractive visualiser, based on all the annotations collected, would increase the reward after text-curation enormously. It would provide scientists with a powerful new tool to gain insights in complex biological matters.

To cite Erhardt on this matter: "The interactivity and dynamics of the visual representation are important aspects of information visualisation. Strong techniques enable the user to modify the visualisation in real time, thus affording unparalleled perception of patterns and structural relations in the abstract data in question" (Erhardt 2006).

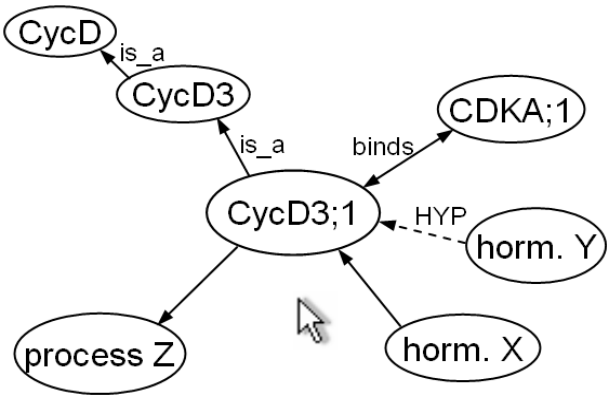


Figure 6.7: Illustration of how an interactive, dynamical information visualiser could graphically represent the information directly associated with CycD3;1. The proteins, processes, etc. that are somehow related to CycD3;1 are connected to it via lines (arrows) that also carry information about the type of relation: is_a, binds, activates, or hypothetical relations, just to show a few. Double-clicking on any visible entity makes it slide to the centre, while its related entities pop out as well.

Based on these considerations, we have developed an interactive, responsive visualiser on top of the manually curated information. It is a prototype reminiscent of graphical browsers like Thinkmap and Visuwords. When one chooses an entity to start with (by typing or selecting in a list), it is centered in the visualiser and all its related entities are shown: see figure 6.7. When the user subsequently double-clicks on any visible entity, that one will slide to the center and its related entities will pop out as well. In this way one can browse through the whole potentially vast resource of relations, while only dealing with manageable view on it at any time. Furthermore one can reorder the node's positions, zoom in or out with the scroll-wheel, and pan (drag) the view window. In summary, our visualiser supports the dynamic exploration of curated information over relations.

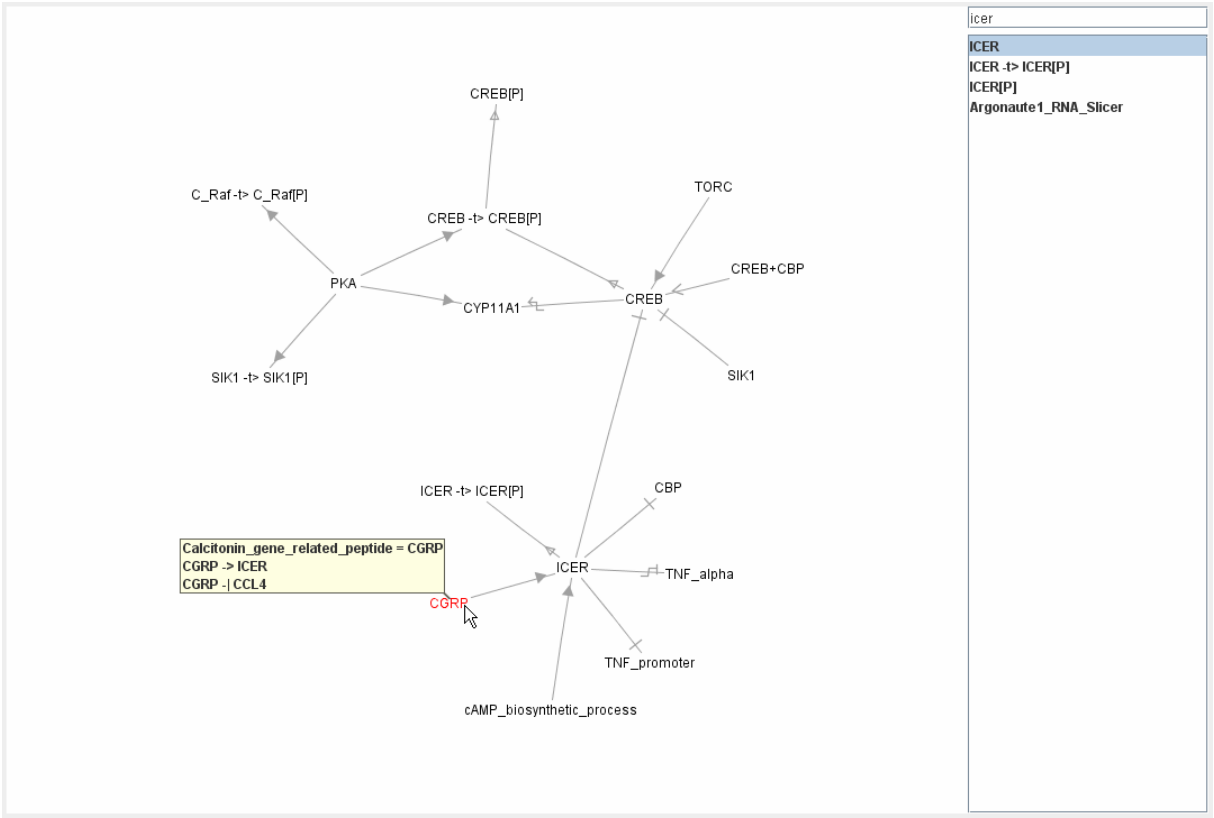


Figure 6.8: Screenshot of MineMap's visualiser. Shown here is some information currently available in MineMap's database. It was manually curated from literature by members of the Gastroenterology group at NTNU, as a practical and introductory group session. For this visualisation, we looked up the protein ICER in the right hand side column (via type and Enter). Then, from the initial visualisation, we also expanded the surroundings of the nodes CREB and PKA. Finally, we placed the mouse cursor on top of the CGRP node until its popup appeared, displaying (in a draft manner) what statements in the database are associated with CGRP. The full arrows stand for activation, the hollow closed ones for participation in a transformation, and lines with a perpendicular dash at the end represent inhibition.

On a little more technical level: we programmed the visualiser in Java, starting from the standard available classes (Swing). It shows animated ('dynamical') force-based layout: whenever the graph structure is changed, the nodes will move towards a new relaxation alignment. This is not only fun to watch and play with, it is also visually essential for being

able to keep track of global node relocations, whenever nodes are added or moved (dragged). We were pleased to see that for several dozens of nodes, the animation still works extremely fast on a five years old computer. The force-based layout algorithm works via electrical repulsion between all nodes, and via physical string-based push/pull forces attached to the relationship lines. Furthermore, the nodes are modelled as *bar* charges (of variable width) rather than point charges. This protects the text labels against overlap, as they now have a full-width power to push each other aside. Also, invisible physical strings are sometimes used to further optimize the visualisation's appeal.
We are aware that several more features can be added to this prototype in the future, like: application of filters or filter combinations (like for information type, species, publication, annotator, etc.), and the option to hide a particular nodes and its attached surroundings (which is not as straightforward as it might seem).

The first users of MineMap's interactive, dynamical visualiser very much appreciate the flexibility to browse through the association and interaction graphs. The information currently in the system is still modest, but one can already experience the visualiser's workings online, at http://www.biology2.net; user = guest, pass = guest. Figure 6.8 already shows a screenshot.

6.6.2 Relation extraction

In fact, section 6.6.1 only describes our stand-alone graph layout and browsing algorithm. But in order to make such visualisations possible in MineMap, we first have to extract a list of nodes, relationships, and relationship types from the syntax tree (MineMap's internal data format from section 6.4). These three things are precisely (and only) what the visualiser needs.

We programmed a module according to the *Design Pattern* "TreeVisitor". (A design pattern is a best-practice programming method). While this algorithm "walks" through our syntax tree, it collects all entities along the way, and it constructs a list of relationships between them. For example for figure 6.5, the three nodes for "A", "B" and "G1.begin" would be extracted, as well as a ternary relation connecting them. The ternary relationship's types would be "no arrow" towards the activator A, "arrow" towards the activee B, and "@-relation" for the temporal part. After that, this list of nodes, relations and types are passed to the visualiser. Note that this tree traversal tends to be simple in many cases, but it can get quite complicated given the full combinatorial power of the language.

6.7 Aspect 4: Web environment for cooperation

So far we have discussed a whole pipeline of modules: the controlled language, the dictionary lookup, the parser, the relation extractor, and the visualiser. Now, in order to make all this functionality available for a cooperative annotation effort, this aggregate program has to be moved online.

Web environment as Java-applet

Ideally, one would now switch gears and make a squeaky slick web-application with PHP, Ajax and MySQL, following the present-day example of the many *Web 2.0* (cooperativity) sites that are appearing like mushrooms after the rain. But also, one would then typically hire a couple of programmers for a year or two to tackle all the design and programming issues that come along with such a project. However, not having these resources at our disposal at this time, and remembering that for now, we have been building a prototype, a demonstration in fact, we had to be more modest. We have instead extended our initial Java software, and built our web-application as a natural Java-applet around it. Still, the result is quite usable for a start.

Using the web environment

Our annotation environment is now an in-browser, web-connected, cooperative software platform, located at http://www.biology2.net . When one navigates to that URL, one first sees a login screen (use guest / guest). After that, one can click a button to choose for either annotating or visualising information. The annotations happen *publication-centered*, i.e. there is one shared annotation (list of statements) for each publication. This resembles the subject-centred setup of Wikipedia, the world's largest, cooperatively built encyclopaedia. To start or continue an annotation in MineMap, one can select a paper based on its unique PubMed identifier (found on pubmed.org), which the applet then uses to query the PubMed site for details. A publication's title, authors, journal and year are automatically stored as metadata next to the annotation.

It is known that in a cooperative environment, one will always have to protect against abuses, since anyone (with a valid login) can add or change the contents of the site (Priedhorsky 2005). Therefore, as a first step in this direction, our platform already stores a back-traceable track of relative changes (a "diff") that were made to reach a publication's current annotation. This diff supports tracking the creation or change of any statement to a particular annotator. This forms important meta-information, and it is also why users should log in to the system.

After one finishes an annotation, a click on the "Save" button makes MineMap check the statements' consistency, report eventual errors, or write them into the database. The visualiser module, fully embedded in the applet and accessible via the main menu, can then build its overview based on all nodes and relations as they appear in the database.

The MySQL database

All annotations, the back-tracing diff-data, user data, and more, are stored in a MySQL database that is accessed via bridging PHP-pages located on our web server. For example when the visualiser needs to retrieve statement information from the database, it addresses a PHP-page on our server (password-protected), similar to how a full PHP-application would use Ajax. The database schema that we conceived is a pragmatic, simple and workable first structure, capable for decent statement storage and retrieval, and visualiser support.

6.8 Use cases

In many of the previous sections in this chapter, we already pointed out various advantages for using MineMap, a cooperation-based biomedical literature annotation system. In this section, we will sum up some examples of how the system can already immediately be used for some concrete applications.

6.8.1 Use cases: Information management

Example 1
The most elementary (and initial) use-case for MineMap is that of a single researcher using it. When she reads her favourite papers, at the same time she uses MineMap as her note-book. Days or months later when she tries to find back a certain fact, instead of re-reading those articles, she goes to her online MineMap-notes and overviews the information there. Over time she can summarize dozens of publications. At that time, the benefits of using a structured format comes to surface. Now she won't have to deal with long pages filled with unstructured notes; instead she will use MineMap's search-and-explore functionality to retrieve the information, explore its context (located between the other statements in an annotation), and explore its connectivity in an interactive diagram (the visualiser).

Example 2
If this same task is now shared by a group of scientists in the same research domain, then the *network effect* appears. A large reading task can be divided over a group of people, of which each reads a manageable part, and each contributes a piece to the growing aggregate result. MineMap's visualiser then combines these facts, coming from an amount of papers that no single person could have read on his own in a reasonable time. Expectedly people will easily encounter new facts uploaded by others that they would otherwise never have been aware of.

6.8.2 Use cases: Exploring the composite information

By the time a cooperatively built knowledge base reaches critical mass, it will also become useful to parts of the public that do not wish to annotate, and that only will use it as an information exploration reference.

Example 1
Suppose that in the genomic screening of a biological experiment, the gene CycD3;1 shows some conspicuous behaviour, and that one wants to learn more about CycD3;1. Instead of (or next to) consulting for instance Gene Ontology to get a general functional description of CycD3;1, one may be interested to learn more details. Typically one would then go to the MineMap site and look up all information concerning this gene. MineMap will surround this gene/protein with all its molecular interactors (like inhibitors, proteins, or hormones), its transgenically induced effects on the phenotype, its involvement in bioprocesses, and so on. This can give a significantly deeper view on the workings of CycD3;1.

Moreover, if a large community supports this manual curation organisation, then one can also expect that newly published information about CycD3;1 will rather quickly be updated in the knowledge repository.

Example 2
One may be studying the G1/S cell cycle checkpoint. This can be viewed in MineMap by looking up the G1_S related interactions (connected to the G1_S node). Although this is still a topic on our to-do-list, one may also apply a filter in the visualiser, for example to see only molecular activation or inhibition relations.

Example 3
As another future feature, it is possible to link an interaction or statement displayed in the visualiser, to the annotation or publication where the statement originates from. In fact, MineMap may be used as a human-reviewed information retrieval tool to find relevant publications that describe a specific topic or interaction. In addition, this could be used to verify if a certain relation was already investigated earlier, before one sets up an experiment of his own.

Example 4
Someone who investigates a hormone_X will be interested to know as many as possible of its target processes or genes. In addition, if the visualiser would be given the ability to overlay genes across organisms, then apparent gaps may generate hypotheses for hormone_X's effect in a certain species.

Example 5
The next example is that of a student who wants to learn more about a biological topic. A freely browsable interaction diagram (the visualiser) can make up a good tool to discover the general structure of a biological process. Also, it can bring up less well-known interactions (e.g. a recent discovery, or a previously inconspicuous publication). Moreover, when every relation is supported by a link back to the publication(s) that describe it, it MineMap can have certain value as a learning tool.

6.8.3 Use cases: Linking with external applications

Example 1
As touched upon before, manually extracted information can serve as an ever-growing set of learning examples to train text-mining algorithms. Note that MineMap does not require from the annotator to mark the position in the text where each statement came from. This is to keep the annotation process as hassle-free as possible.
Still, based on the MineMap information repository, a text-miner could try to extract all the statements that the human annotator extracted, from anywhere in the text. This exercise is in fact similar to the BioCreAtIvE challenge, task 2 (Blaschke 2005), where participants had to automatically relate proteins to a Gene Ontology category, based on merely the full-text of an article.

Example 2

Finally, MineMap could also combine its manually curated information with that from other, institutional manual curation efforts, like GO or KEGG; and visualise one composite overview.

6.9 First practical sessions & feedback

In the use-case section, we spoke from the perspective that the MineMap knowledge base could one day contain a substantial amount of information. Currently, however, we are only at the beginning. Our accomplishments so far, have been to design this concept, work out a prototype, gather some initial interest, and from that, organize a number of first test sessions to receive feedback and fresh perspective.

We have established contact with three different groups of interested people, who we are truly thankful for participating in the first test sessions, and for instructive interactions and comments. The first group was an internal group in our department, in conjunction with the Agronomics project. Insights developed from this interaction have boosted MineMap's initial development in several respects (among which the web-based cooperativity and the dictionary lookup needs), and we owe our thanks especially to Fabio Fiorani and Pierre Hilson for this opportunity. However, MineMap could not yet fulfil all of their requirements to cover transgenic-line definitions with substantially more detail and variety, on the desired short notice. MineMap, in its fledgling phase, has shown to be still more oriented towards molecular interaction details and network modelling for now.
The test session with a second group, the Gastroenterology group at NTNU, was a positive experience where participants were able to capture various protein interactions from literature (the type of information that MineMap is currently still best suited for). Part of this result was already shown in figure 6.8. This was also the first time when our web-based application and the dictionary lookup service were both operational.
The third group is an HSFP project, in particular represented by Ewa Sugajska and Jens Hollunder. This interaction is still in progress at the time of writing, and it has by now resulted in again various new insights concerning future extensions for the vocabulary.

6.10 Further thoughts on the MineMap system

What to extract
Although during the design of the controlled language, we had taken ten cell cycle review papers and completely annotated them, this is not necessarily how all MineMap users will behave. We witnessed that several people rather like to take a stack of publications and hunt for specific information types in all of them; for example, everything pertaining to cell growth control. As a result, one may expect that next to fully-annotated articles, there will be many "stub"s in MineMap as well (like the Wikipedia class of too short articles). This is not really a problem, since other users of MineMap can add more information to it afterwards, if they wish. However, since a biological publication is finite in length (in contrast to a Wikipedia article, which can always grow some more), one should probably somehow signal the completion of a full annotation, to 'close the topic'.

Still, at what point an annotation is complete, could depend on the annotator's judgement. In that respect, it can be interesting to let different persons annotate the same paper, and to use these differences as lessons for defining annotation guidelines.

Copyright considerations
One may be worried about copyright issues. However, only interpretations (translations) of the text from an article are used; the text itself is not copied. And then still, if the original sentence would be stored as extra info, still this would only be a small quotation.
Actually, copyright holders will probably even welcome the idea of manually curated text-annotations. Since extracted statements would in fact act as pointers to the annotated publications, this will make them easier to find and thus more often requested.

Structured abstracts as annotations
The best quality translation of a publication's facts into the structured format can probably be made the authors themselves, or alternatively by volunteers who are genuinely interested in the topic. We believe that authors could be encouraged to compose a structured abstract of their main findings, next to the full-text abstract. Whether this information is given as a supplement to their publication, or put immediately in a shared system like MineMap, in any case the authors will benefit from this computer-readability by gaining a higher visibility.

If you have ideas, you have the main asset you need, and there isn't any limit to what you can do with your business and your life. Ideas are any man's greatest asset.
- Harvey S. Firestone

6.11 Future perspectives

The novel concept and the multi-modular application we described in the previous sections stands for a considerably ambitious project. By launching prototypic software, we already acquired hands-on experience and were able to locate and solve some of the main usability issues. Also, the extensive feedback that we received and the time we spent to develop our ideas further towards future possibilities, resulted in a sizable list of possible extensions, modifications and suggestions, a few of which were already mentioned briefly. Considering a project of this complexity, and moreover with a single person developing the prototype and having a PhD thesis expected due, it is important first to reconsider and evaluate things, before heeding over to the next level. In the previous section, a number of essential upgrades to the initial product were already described (term lookup assistance and web functionality), but with the new insights we have collected now, we realize that a thorough rewrite of the system will be necessary in order to bring it to the second level. Such an effort is not realizable within the confinements of one PhD project. However, the up-to-now gathered insights towards future development do comprise a valuable extra part of our research, so we will describe some of them briefly in what follows.

6.11.1 Syntax extensions

Some of the ideas that we formed during our applications, for upgrades of the controlled language:

- More detailed hypothesis specification (in percent or as fuzzy variable).
- Relation modifiers, like "A ->[direct] B" to specify that it is a direct interaction and not an indirect one. The statement "A -> B" would then still simply stand for unspecified stimulation. Sometimes this information is also not given in a publication anyway (like for a review). On the meta-level, "->" could be a parent of both "->[direct]" and "->[indirect]", just like "to move" would be a parent of "to walk" and "to run". This is similar to our existing concept of quantification modifiers, like "é[many]".
- For protein binding: protein complexes could be specifiable with a binding-operator "^", like in "A^B^C". Also subcomplexes could be specifiable, like in "(A^B)^C".
- Allowing the use of ontology terms for describing relations, next to the existing symbolic operators. This would permit representing the widest spectrum of relations in our language. It would also have repercussions on the relation extractor, as it should be told how to deal with new relations.
- Several provisions for more detailed transgenic line specifications. As one example, the description of a loss-of-function mutant having a Single Nucleotide Polymorphism at position X on a gene, could look like "Arath[Gene1=LOF[SNP,posX]]". To accommodate this, our language should among others be extended to allow operators inside modifiers as well recursive modifiers.
- More powerful ontology support by inconspicuously managing ontology terms by their ID.

6.11.2 Further Input Assistant development

We have already built an input assistant module as a text pane with an inline ontology/dictionary lookup service and convenient term-autocompletion. The module could even more facilitate the annotation experience in the future:

- Syntax highlighting: showing the operators in a different color would emphasize a statement's structure, making it easier and quicker to read; just like in SIM-plex.
- Next to term suggestion, also operator suggestion: announcing which valid symbol may follow at any point. This would require the input assistant to become syntax-aware, and stay up-to-date with the parser through various syntax upgrades.
- Composing frequently reoccurring statements via fixed text-fields, in a graphical user interface layer on top that generates textual statements. Custom templates to facilitate user generation of such user interface modules.
- Literature mining algorithms could already suggest a statement, which could speeden up manual annotation, depending on the quality of the algorithms. Or it could already let the autocomplete module know which terms to expect.

6.11.3 Further Visualiser development

The attractive, interactively browsable visualiser that we built offers basic but solid functionality. Many extensions are possible, among which:

- All kinds of decorations for relations or nodes, like colours (activation/inhibition), different shapes for different node types (based on type information provided by the dictionary).
- Filters: to leave out, or only show, certain types of nodes or relations. For example to exclude hypothesised relations, or to view only activation and inhibition relations in order to obtain a common gene network representation.
- Links from the visualiser to a statement's source annotation or publication.
- User-friendly listing of a node's associated statements, in a list next to the graphic area.
- Technical upgrade: making the relation extractor scalable to a large database (the current working solution loads all statements in memory). Relations could already be calculated at annotation submission, stored in the database, and used in fragment-based delivery of information to the visualiser.

6.11.4 Data storage design extensions

- Store relation extractor's results in the database for visualiser efficiency.
- Support reference to various kinds of information sources: not only articles but also books or not (yet) published information. Measures would be needed to keep the latter manageable and clean.

6.11.5 Web application design

- Transition from Java-applet to full web application, e.g. programmed in PHP/JavaScript. Page-based, flexibility, appealing design. Improved user experience via a full 'Web 2.0' (cooperative site) look-and-feel. Several parts of MineMap would be rewritten.
- Taking into account the importance of a globally appealing design; as learned from the interest that people showed after just seeing MineMap's attractive visualiser.
- User management and user community handling. Registration, profile, access level or power level (rookie, moderator, administrator).
- Enhanced publication lookup support. Facility to filter by annotator or annotator group.

6.11.6 Information export

In addition to providing information visualisation as one of the applications of a potentially large collectively-built data repository, some other promising opportunities to benefit from this initiative can be thought of too.

- Export to a computer-oriented information representation format such as the Web Ontology Language (OWL). OWL sets out to be a universally usable format to describe and store data with rich semantics attached to it. This would allow *reasoner* algorithms to evaluate the curated information. They could infer new hypotheses or, more elementary, they could search for inconsistencies between two publications. For example if one publication suggests that protein A and B physically bind, but another one says they have no binding site and thus can not bind, a contradiction can be inferred.

- Application as training set for automated text-mining algorithms, as already mentioned earlier, in section 6.8.3.
- Integration with other information or data repositories, to create an aggregate overview.

6.12 The heuristic solution called MineMap

A *heuristic approach* is one that begins with an approximate method of solving a complex problem within the context of a goal, and then uses feedback from evaluating that solution in order to further improve it.

In that respect, MineMap represents a heuristic approach to tackle the intricate problem of biological literature annotation. In order to augment the information access experience of biologists, we have heuristically researched the novel concept of manual text-curation linked to an attractive reward, in combination with the power of web-based cooperativity. MineMap represents a first, concrete solution towards that bold goal, and both yielded a working information collection platform, as well as plenty of ideas for the future.

Whatever you do will be insignificant,
but it is very important that you do it.
- Mahatma Gandhi

Chapter 7

Biology 2.0:
A Network of Knowledge

In the previous chapters, we have learned some basic lessons about cooperativity and manual text curation, thanks to our experience with MineMap. Now we wish to convey this *concept* of community-based manual curation to the general public. Therefore we have prepared an article which will be submitted soon, and which we have included as a Chapter 7 in this thesis. It presents the idea of the merger between the "*Web 2.0*" concept (web-based cooperative projects) and the domain of Biology, hence the title "Biology 2.0".

The problem: Retrieving information from scientific literature

New scientific findings are mainly shared through publications in research journals. Paradoxically, by publishing papers a scientist unwillingly hides information in a format that can hardly be understood by computers: natural language. Finding information in scientific literature requires careful reading of selected publications, a task essential for any scientist but becoming increasingly difficult given the accelerating growth of scientific literature. Biologists would be thrilled if they could get an integrated view on the information in literature, just a few mouse-clicks away. Automated text mining approaches are aimed at this, but accurate and complete information extraction from literature is only in its infancy. Therefore we propose to launch a community-wide effort in manual text curation, converting progressively the most important scientific papers into a structured format that can be understood both by computers and people. Only a community of enthusiastic biologists has the power to achieve this large task.

The solution: Core concept

We present the concept of a Wikipedia-style resource for highly structured biological information. This would be a place where any biologist can add accurate, detailed and diverse information that describes all scientific facts from publications, like "protein A activates gene B at time T in cell-type C". Here, any biologist may subsequently benefit from the integrative and accumulative effect of community-based efforts (figure 8.1), like we are witnessing today in many user-content built websites.

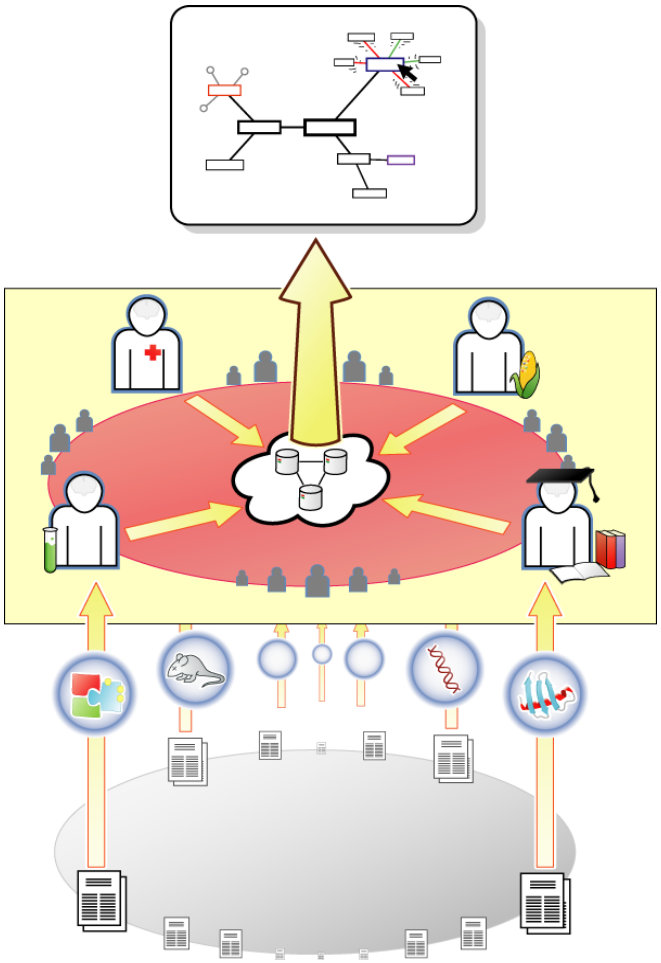


Figure 8.1. The cooperative network of 'Biology 2.0'. Bottom: Large numbers of biological publications supply a plethora of diverse information (like protein complexes, mutant phenotypes, gene activation and protein domains). Middle: Most of this enormous amount of information is designed to be interpreted properly only by people, so biologists from various research fields (like medicine, plant or animal, or even students) should work together to feed simple, ready pieces into a central resource, a Wikipedia-like ('Web 2.0') website. Top: From this resource, algorithms can compose an integrated view, such as a relationship diagram, interactively browsable over relation-links. All: Much biological information, especially where it describes relationships and interactions, is now hidden in literature. In a 'Biology 2.0' set-up, as we coin it, a community effort can convert this information to structured facts that can be integrated and shared with all.

Biologists hide facts in natural language

In order to write good scientific publications, scientists work hard to translate a state-of-the-art, their new results, and logically inferred conclusions into an appealing natural language format. Ideally this knowledge, embedded in scientific papers, should be retrievable easily upon launching a simple query in a text miner. The biosciences, however, pose a tremendous challenge to automated text mining (Erhardt 2006, Jensen 2006, Blaschke 2005), as scientific publications are packed with highly ambiguous phrases that are often embedded in complex relationships. To make matters worse, a correct interpretation of the information may depend on significant biological background knowledge, or an understanding of the particular textual context. Text mining may have an acceptable performance in some fields but it leaves much to be desired in the biosciences. As a result, the majority of the knowledge that resides in biological literature can only be interpreted properly by human intellect, and can only be extracted properly through human intervention.

A simple annotation system

In order to condense information from literature, one requires a common, simplified and structured language, a language that biologists can write without too much training, and that computers can understand without ambiguity. Still, it should capture a wide variety of information types, and be flexible enough to compose many specific details. For our own efforts in modelling cell cycle control, we have developed a prototypic annotation system

that had to surpass other text curation systems (Kim 2003, Kuhn 2006, Racunas 2004) in its emphasis on expression power (for examples, see: www.biology2.net). The textual notation combines the intuitiveness of graphical protein interaction diagram notations (Kohn 2006, Kitano 2005) with a wide variety of other information types, like temporal-spatial information, quantities and relations, mutant phenotypes, and even hypotheses. We anticipate, however, that the full development of such a notation will be a long-term process with many iterative improvements, to meet the needs of scientific communities in various biological subfields.

Beating Babel with dictionaries

Combining information extracted by various people, however, is only possible if there exists a common vocabulary. Otherwise, one person would talk about e.g. 'cat', while others would use a term like 'felis'. To avoid such Babylonian confusion, biologists from several disciplines are building ontologies. Ontologies can be seen as topic-specific dictionaries with well-defined, unique terms for many concepts, such as molecular functions, interaction types, or organ structures (Ashburner 2000, Jaiswal 2005, Ilic 2007). In our prototype application, we discovered that technical facilities like in-line lookup of ontology terms and auto-completion of synonyms with their preferred terms, are essential for a smooth user experience. Our test sessions also identified several terms not yet covered by ontologies. So a lookup service could also suggest new terms that are community-defined, or even terms based on automated text-mining (Krallinger 2005). This shows that the emerging ontologies will also benefit from a large-scale use, and can be furthered by community discussion on ontology terms.

The cooperative "Biology 2.0" organization

When many biologists use the same notation to condense knowledge from literature, their gathered information can all be added to a shared, publicly accessible resource, where it can be reviewed and commented upon by others. In essence, this is what biology needs: a cooperative effort, a so-called "Web 2.0" organisation. The Wikipedia encyclopaedia brilliantly illustrates this Web 2.0 principle. Here, control for information gathering and structuring is given entirely to its community of users, and the cumulative value increases tremendously as more and more people contribute. Such a large-scale collaborative project will also be essential to salvage the information scattered over piles of biological publications. But with a major difference: the biological information must be made structured enough to let computers understand and integrate it. Therefore, a so-called 'Biology 2.0' community-driven cooperation has to start small. It needs individual biologists willing to make test annotations of their favourite information, and use an experimental, shared language that can grow only with diverse real-user feedback. Whether this happens for selected full-paper annotations, or with more feasible, target-oriented partial annotations, these small-scale initiatives will also be seminal for further large-scale outgrowth, where biologists can together create a huge network of knowledge.

Role of journals

No doubt a community of biologists should be able to manually curate significant sections of literature covering their favourite research domains. However, it should be feasible to tackle the problem of text curation at the source. Authors of newly submitted manuscripts should be encouraged, or even required to provide a supplementary data file with the main

new results and hypotheses, as a structured list of statements. These listed facts would outline the essence of the manuscript, but in a computer readable form that allows them to be immediately added to a common knowledge repository. Such a supplementary fact sheet would in no way take away the wish to read the full paper. Actually, this partial open access to scientific knowledge may result in higher visibility, and the attention might be directed to specific literature that can then be examined more closely. If journal editors would enforce this, it could greatly boost the growth of Biology 2.0.

The immediate reward factor

An immediate reward for this annotation effort is vital. Only when biologists get a powerful and direct benefit, they will be happy to go the extra mile for text annotation. Therefore we built an interactive visualiser prototype that can show any biological concept from the integrated information resource, and that dynamically lays out its closely related concepts (figure 1, top; figure 2). Much like how the human brain works, it then allows users to hop from one concept to related ones, each time reorganizing the graph in an appealing, dynamical way (look & feel: see www.biology2.net). Like this, people's newly entered facts are directly integrated with those of fellow scientists into a formidable knowledge resource that everyone can explore via an intuitive, attractive access tool. We have witnessed that this immediate reward for the human effort is vital and creates a significant incentive to make contributions in literature curation.

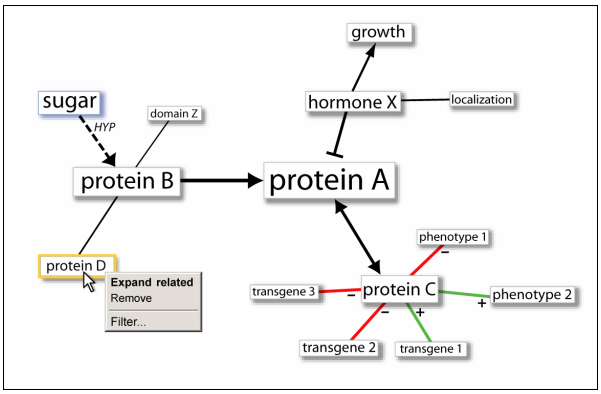


Figure 2. Browsing shared information. Sketch of an interactive visualisation that integrates all information gathered by the 'Biology 2.0' community effort. Various entities are depicted in boxes, diverse relationships between them with lines and arrows.

Computing knowledge

Structured information in generally accepted ontology terms can also be exported to the Web Ontology Language (OWL). Data in the OWL format is semantically rich and can be used by reasoning algorithms that check Biology 2.0 data consistency, or that synthesize information into new hypotheses. It opens the door for computational approaches to investigate the growing stream of facts presented by literature. A structured Biology 2.0 information resource may become crucial for systems biology.

Summary

In summary, we pointed out the necessity and lined out the requirements for a concerted effort in biological literature annotation. In order to utilize all the information scattered over publications, a cooperative 'Biology 2.0' organisation demands a number of building blocks. These include a simple language structure that is manageable both for people and computers, a common vocabulary, a central site for biologist cooperation, and an

attractive reward for the human effort. We have experienced these requisites based on our prototype software (www.biology2.net), and find it time to spread the Biology 2.0 message. Input from various biological disciplines is now needed to further refine a common language structure, plus a willingness to translate both existing and new findings into this structured format. A broad community-based Biology 2.0 organization may constitute a critical evolution for systems biology. The conversion of scientific results into a common annotation format, shared online, offers the perspective to migrate a vast body of highly dispersed literature facts into a powerful, integrated net of knowledge.

References

Chapter 1 references

Beemster GT, De Veylder L, Vercruysse S, West G, Rombaut D, Van Hummelen P, Galichet A, Gruissem W, Inzé D, Vuylsteke M. Genome-wide analysis of gene expression profiles associated with cell cycle transitions in growing organs of Arabidopsis. Plant Physiol. 2005 Jun;138(2):734-43.

Beemster GT, Vercruysse S, De Veylder L, Kuiper M, Inzé D. The Arabidopsis leaf as a model system for investigating the role of cell cycle regulation in organ growth. J Plant Res. 2006 Jan;119(1):43-50.

Blaschke C, Leon EA, Krallinger M, Valencia A. Evaluation of BioCreAtIvE assessment of task 2. BMC Bioinformatics. 2005;6 Suppl 1:S16.

Glass L, Kauffman SA. The logical analysis of continuous, non-linear biochemical control networks. J Theor Biol. 1973 Apr;39(1):103-29.

Hilson P, Allemeersch J, Altmann T, Aubourg S, Avon A, Beynon J, Bhalerao RP, Bitton F, Caboche M, Cannoot B, Chardakov V, Cognet-Holliger C, Colot V, Crowe M, Darimont C, Durinck S, Eickhoff H, de Longevialle AF, Farmer EE, Grant M, Kuiper MT, Lehrach H, Léon C, Leyva A, Lundeberg J, Lurin C, Moreau Y, Nietfeld W, Paz-Ares J, Reymond P, Rouzé P, Sandberg G, Segura MD, Serizet C, Tabrett A, Taconnat L, Thareau V, Van Hummelen P, Vercruysse S, Vuylsteke M, Weingartner M, Weisbeek PJ, Wirta V, Wittink FR, Zabeau M, Small I. Versatile gene-specific sequence tags for Arabidopsis functional genomics: transcript profiling and reverse genetics applications. Genome Res. 2004 Oct;14(10B):2176-89.

Himanen K, Vuylsteke M, Vanneste S, Vercruysse S, Boucheron E, Alard P, Chriqui D, Van Montagu M, Inzé D, Beeckman T. Transcript profiling of early lateral root initiation. Proc Natl Acad Sci U S A. 2004 Apr 6;101(14):5146-51.

Hosack DA, Dennis G Jr, Sherman BT, Lane HC, Lempicki RA. Identifying biological themes within lists of genes with EASE. Genome Biol. 2003;4(10):R70.

Kitano H, Funahashi A, Matsuoka Y, Oda K. Using process diagrams for the graphical representation of biological networks. Nat Biotechnol. 2005 Aug;23(8):961-6.

Kohn KW. Molecular interaction maps as information organizers and simulation guides. Chaos. 2001 Mar;11(1):84-97.

Novak B, Pataki Z, Ciliberto A, Tyson JJ. Mathematical model of the cell division cycle of fission yeast. Chaos. 2001 Mar;11(1):277-286.

Proust, Remembrance of Things Past, vol.1: Swann's Way (À la recherche du temps perdu). 1913-27.

Rual JF, Venkatesan K, Hao T, Hirozane-Kishikawa T, Dricot A, Li N, Berriz GF, Gibbons FD, Dreze M, Ayivi-Guedehoussou N, Klitgord N, Simon C, Boxem M, Milstein S, Rosenberg J, Goldberg DS, Zhang LV, Wong SL, Franklin G, Li S, Albala JS, Lim J, Fraughton C, Llamosas E, Cevik S, Bex C, Lamesch P, Sikorski RS, Vandenhaute J, Zoghbi HY, Smolyar A, Bosak S, Sequerra R, Doucette-Stamm L, Cusick ME, Hill DE, Roth FP, Vidal M. Towards a proteome-scale map of the human protein-protein interaction network. Nature. 2005 Oct 20;437(7062):1173-8.

Saeed AI, Sharov V, White J, Li J, Liang W, Bhagabati N, Braisted J, Klapa M, Currier T, Thiagarajan M, Sturn A, Snuffin M, Rezantsev A, Popov D, Ryltsov A, Kostukovich E, Borisovsky I, Liu Z, Vinsavich A, Trush V, Quackenbush J. TM4: a free, open-source system for microarray data management and analysis. Biotechniques. 2003 Feb;34(2):374-8.

Vercruysse S, Kuiper M. Simulating genetic networks made easy: network construction with simple building blocks. Bioinformatics. 2005 Jan 15;21(2):269-71.

Verkest A, Manes CL, Vercruysse S, Maes S, Van Der Schueren E, Beeckman T, Genschik P, Kuiper M, Inzé D, De Veylder L. The cyclin-dependent kinase inhibitor KRP2 controls the onset of the endoreduplication cycle during Arabidopsis leaf development through inhibition of mitotic CDKA;1 kinase complexes. Plant Cell. 2005 Jun;17(6):1723-36.

Chapter 2 references

Adalsteinsson D, McMillen D, Elston TC. Biochemical Network Stochastic Simulator (BioNetS): software for stochastic modeling of biochemical networks. BMC Bioinformatics. 2004 Mar 8;5:24.

Batt G, Ropers D, de Jong H, Geiselmann J, Mateescu R, Page M, Schneider D. Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in Escherichia coli. Bioinformatics. 2005 Jun;21 Suppl 1:i19-28.

Briggs GE, Haldane JBS. A note on the kinetics of enzyme action. Biochem. J. 1925. 19:339-339.

Casey R, de Jong H, Gouzé JL. Piecewise-linear models of genetic regulatory networks: equilibria and their stability. J Math Biol. 2006 Jan;52(1):27-56.

Chaouiya C. Petri net modelling of biological networks. Brief Bioinform. 2007 Jul;8(4):210-9.

Chen KC, Calzone L, Csikasz-Nagy A, Cross FR, Novak B, Tyson JJ. Integrative analysis of cell cycle control in budding yeast. Mol Biol Cell. 2004 Aug;15(8):3841-62.

Csikász-Nagy A, Battogtokh D, Chen KC, Novák B, Tyson JJ. Analysis of a generic model of eukaryotic cell-cycle regulation. Biophys J. 2006 Jun 15;90(12):4361-79.

Davuluri RV, Sun H, Palaniswamy SK, Matthews N, Molina C, Kurtz M, Grotewold E. AGRIS: Arabidopsis gene regulatory information server, an information resource of Arabidopsis cis-regulatory elements and transcription factors. BMC Bioinformatics. 2003 Jun 23;4:25.

de Jong H. Modeling and simulation of genetic regulatory systems: a literature review. J Comput Biol. 2002;9(1):67-103.

de Jong H, Geiselmann J, Hernandez C, Page M. Genetic Network Analyzer: qualitative simulation of genetic regulatory networks. Bioinformatics. 2003 Feb 12;19(3):336-44.

de Jong H, Gouzé JL, Hernandez C, Page M, Sari T, Geiselmann J. Qualitative simulation of genetic regulatory networks using piecewise-linear models. Bull Math Biol. 2004 Mar;66(2):301-40.

Devloo V, Hansen P, Labbé M. Identification of all steady states in large networks by logical analysis. Bull Math Biol. 2003 Nov;65(6):1025-51.

D'haeseleer P, Liang S, Somogyi R. Genetic network inference: from co-expression clustering to reverse engineering. Bioinformatics. 2000 Aug;16(8):707-26.

Edwards R, van den Driessche P, Wang L. Periodicity in piecewise-linear switching networks with delay. J Math Biol. 2007 Aug;55(2):271-98.

Friedman N, Linial M, Nachman I, Pe'er D. Using Bayesian networks to analyze expression data. J Comput Biol. 2000;7(3-4):601-20.

Gibson MA, Mjolsness E. Modeling the activity of single genes. In Computational Modeling of Genetic and Biochemical Networks. Bower JM, Bolouri H, editors. MIT Press, Cambridge, MA. 2001. 3-48.

Gierer A, Meinhardt H. A theory of biological pattern formation. Kybernetik. 1972 Dec;12(1):30-9.

Glass L, Kauffman SA. The logical analysis of continuous, non-linear biochemical control networks. J Theor Biol. 1973 Apr;39(1):103-29.

Goldbeter A. Computational approaches to cellular rhythms. Nature. 2002 Nov 14;420(6912):238-45.

Goryanin I, Hodgman TC, Selkov E. Mathematical simulation and analysis of cellular metabolism and regulation. Bioinformatics. 1999 Sep;15(9):749-58.

Goss PJ, Peccoud J. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. Proc Natl Acad Sci U S A. 1998 Jun 9;95(12):6750-5.

Gouzé JL, Sari T. A class of piecewise linear differential equations arising in biological models. 2003 Dyn Syst. 17:299-316.

Hartwell LH, Hopfield JJ, Leibler S, Murray AW. From molecular to modular cell biology. Nature. 1999 Dec 2;402(6761 Suppl):C47-52.

Holloway DM, Harrison LG. Pattern Selection in Plants: Coupling Chemical Dynamics to Surface Growth in Three Dimensions. Ann Bot (Lond). 2007 Nov 28 [Epub ahead of print]

Ideker T, Lauffenburger D. Building with a scaffold: emerging strategies for high- to low-level cellular modeling. Trends Biotechnol. 2003 Jun;21(6):255-62.

Kaandorp JA, Sloot PM. Morphological models of radiate accretive growth and the influence of hydrodynamics. J Theor Biol. 2001 Apr 7;209(3):257-74.

Kauffman SA. Metabolic stability and epigenesis in randomly constructed genetic nets. J Theor Biol. 1969 Mar;22(3):437-67.

Kauffman SA. Antichaos and adaptation. Sci Am. 1991 Aug;265(2):78-84.

Kim SY, Imoto S, Miyano S. Inferring gene networks from time series microarray data using dynamic Bayesian networks. Brief Bioinform. 2003 Sep;4(3):228-35.

Kitano H. The standard graphical notation for biological networks. The Sixth Workshop on Software Platforms for Systems Biology. 2002.

Kitano H, Funahashi A, Matsuoka Y, Oda K. Using process diagrams for the graphical representation of biological networks. Nat Biotechnol. 2005 Aug;23(8):961-6.

Kohn KW. Functional capabilities of molecular network components controlling the mammalian G1/S cell cycle phase transition. Oncogene. 1998 Feb 26;16(8):1065-75.

Kohn KW. Molecular interaction maps as information organizers and simulation guides. Chaos. 2001 Mar;11(1):84-97.

Kohn KW, Pommier Y. Molecular interaction map of the p53 and Mdm2 logic elements, which control the Off-On switch of p53 in response to DNA damage. Biochem Biophys Res Commun. 2005 Jun 10;331(3):816-27.

Kohn KW, Aladjem MI, Weinstein JN, Pommier Y. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. Mol Biol Cell. 2006 Jan;17(1):1-13.

Li H, Xuan J, Wang Y, Zhan M. Inferring regulatory networks. Front Biosci. 2008 Jan 1;13:263-75.

Machné R, Finney A, Müller S, Lu J, Widder S, Flamm C. The SBML ODE Solver Library: a native API for symbolic and fast numerical analysis of reaction networks. Bioinformatics. 2006 Jun 1;22(11):1406-7.

Matsuno H, Doi A, Nagasaki M, Miyano S. Hybrid Petri net representation of gene regulatory network. Pac Symp Biocomput. 2000;:341-52.

McAdams HH, Shapiro L. Circuit simulation of genetic networks. Science. 1995 Aug 4;269(5224):650-6.

McAdams HH, Arkin A. Stochastic mechanisms in gene expression. Proc Natl Acad Sci USA. 1997 Feb 4;94(3):814-9.

Meinhardt H, Gierer A. Applications of a theory of biological pattern formation based on lateral inhibition. J Cell Sci. 1974 Jul;15(2):321-46.

Meinhardt H. Models of Biological Pattern Formation. 1982 Nov. Academic Press, London. ISBN 978-0124886209.

Mendes P. GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. Comput Appl Biosci. 1993 Oct;9(5):563-71.

Mendoza L, Thieffry D, Alvarez-Buylla ER. Genetic control of flower morphogenesis in Arabidopsis thaliana: a logical analysis. Bioinformatics. 1999 Jul-Aug;15(7-8):593-606.

Merks R, Hoekstra A, Kaandorp J, Sloot P. Models of coral growth: spontaneous branching, compactification and the Laplacian growth assumption. J Theor Biol. 2003 Sep 21;224(2):153-66.

Merks RM, Hoekstra AG, Kaandorp JA, Sloot PM. Polyp oriented modelling of coral growth. J Theor Biol. 2004 Jun 21;228(4):559-76.

Merks RM, Van de Peer Y, Inzé D, Beemster GT. Canalization without flux sensors: a traveling-wave hypothesis. Trends Plant Sci. 2007 Sep;12(9):384-90.

Michaelis L, Menten M. Die Kinetik der Invertinwirkung. Biochem. Z. 1913. 49:333-369.

Michoel T, Maere S, Bonnet E, Joshi A, Saeys Y, Van den Bulcke T, Van Leemput K, van Remortel P, Kuiper M, Marchal K, Van de Peer Y. Validating module network learning algorithms using simulated data. BMC Bioinformatics. 2007 May 3;8 Suppl 2:S5.

Nagasaki M, Doi A, Matsuno H, et al. A versatile Petri net based architecture for modeling and simulation of complex biological processes. Genome Inform 2004;15:180-97.

Novak B, Tyson JJ. Numerical analysis of a comprehensive model of M-phase control in Xenopus oocyte extracts and intact embryos. J Cell Sci. 1993 Dec;106 (Pt 4):1153-68.

Novak B, Pataki Z, Ciliberto A, Tyson JJ. Mathematical model of the cell division cycle of fission yeast. Chaos. 2001 Mar;11(1):277-286.

Novak B, Tyson JJ. A model for restriction point control of the mammalian cell cycle. J Theor Biol. 2004 Oct 21;230(4):563-79.

Ong IM, Glasner JD, Page D. Modelling regulatory pathways in E. coli from time series expression profiles. Bioinformatics. 2002;18 Suppl 1:S241-8.

Pe'er D, Regev A, Elidan G, Friedman N. Inferring subnetworks from perturbed expression profiles. Bioinformatics. 2001;17 Suppl 1:S215-24.

Perrin BE, Ralaivola L, Mazurie A, Bottani S, Mallet J, d'Alché-Buc F. Gene networks inference using dynamic Bayesian networks. Bioinformatics. 2003 Oct;19 Suppl 2:ii138-48.

Petri CA. Kommunikation mit Automaten. Bonn: Institut fur Instrumentelle Mathematik, Schriften des IIM 1962 Nr.3. English translation: Communication with Automata. New York: Griffiss Air Force Base, Tech Rep RADC-TR-65-377 1966;1 Suppl 1:1.

Racunas SA, Shah NH, Albert I, Fedoroff NV. HyBrow: a prototype system for computer-aided hypothesis evaluation. Bioinformatics. 2004 Aug 4;20 Suppl 1:i257-64.

Racunas SA, Shah NH, Fedoroff NV. A case study in pathway knowledgebase verification. BMC Bioinformatics. 2006 Apr 8;7:196.

Rombauts S, Florquin K, Lescot M, Marchal K, Rouzé P, van de Peer Y. Computational approaches to identify promoters and cis-regulatory elements in plant genomes. Plant Physiol. 2003 Jul;132(3):1162-76.

Ruoff P, Vinsjevik M, Monnerjahn C, Rensing L. The Goodwin model: simulating the effect of light pulses on the circadian sporulation rhythm of Neurospora crassa. J Theor Biol. 2001 Mar 7;209(1):29-42.

Salis H, Sotiropoulos V, Kaznessis YN. Multiscale Hy3S: hybrid stochastic simulation for supercomputers. BMC Bioinformatics. 2006 Feb 24;7:93.

Sánchez L, Thieffry D. A logical analysis of the Drosophila gap-gene system. J Theor Biol. 2001 Jul 21;211(2):115-41.

Sánchez L, Thieffry D. Segmenting the fly embryo: a logical analysis of the pair-rule cross-regulatory module. J Theor Biol. 2003 Oct 21;224(4):517-37.

Schlitt T, Brazma A. Current approaches to gene regulatory network modelling. BMC Bioinformatics. 2007 Sep 27;8 Suppl 6:S9.

Shvartsman SY, Muratov CB, Lauffenburger DA. Modeling and computational analysis of EGF receptor-mediated cell communication in Drosophila oogenesis. Development. 2002 Jun;129(11):2577-89.

Simão E, Remy E, Thieffry D, Chaouiya C. Qualitative modelling of regulated metabolic pathways: application to the tryptophan biosynthesis in E.coli. Bioinformatics. 2005 Sep 1;21 Suppl 2:ii190-6.

Smolen P, Baxter DA, Byrne JH. Mathematical modeling of gene networks. Neuron. 2000 Jun;26(3):567-80.

Smolen P, Baxter DA, Byrne JH. Modeling transcriptional control in gene networks -methods, recent results, and future directions. Bull Math Biol. 2000b Mar;62(2):247-92.

Srivastava R, Peterson MS, Bentley WE. Stochastic kinetic analysis of the Escherichia coli stress circuit using sigma(32)-targeted antisense. Biotechnol Bioeng. 2001 Oct 5;75(1):120-9.

Thieffry D, Thomas R. Dynamical behaviour of biological regulatory networks--II. Immunity control in bacteriophage lambda. Bull Math Biol. 1995 Mar;57(2):277-97.

Thomas R, Kaufman M. Multistationarity, the basis of cell differentiation and memory. II. Logical analysis of regulatory networks in terms of feedback circuits. Chaos. 2001 Mar;11(1):180-195.

Troncale S, Tahi F, Campard D, Vannier JP, Guespin J. Modeling and simulation with Hybrid Functional Petri Nets of the role of interleukin-6 in human early haematopoiesis. Pac Symp Biocomput. 2006;:427-38.

Tyson JJ, Othmer, HG. The dynamics of feedback control circuits in biochemical pathways. 1978. Prog. Theor. Biol. 5, 1-62.

Ueda HR, Hagiwara M, Kitano H. Robust oscillations within the interlocked feedback model of Drosophila circadian rhythm. J Theor Biol. 2001 Jun 21;210(4):401-6.

van Kampen NG. 1992. Stochastic Processes in Physics and Chemistry, 2nd Ed. Elsevier, Dordrecht.

Vercruysse S, Kuiper M. Simulating genetic networks made easy: network construction with simple building blocks. Bioinformatics. 2005 Jan 15;21(2):269-71.

Verkest A, Manes CL, Vercruysse S, Maes S, Van Der Schueren E, Beeckman T, Genschik P, Kuiper M, Inzé D, De Veylder L. The cyclin-dependent kinase inhibitor KRP2 controls the onset of the

endoreduplication cycle during Arabidopsis leaf development through inhibition of mitotic CDKA;1 kinase complexes. Plant Cell. 2005 Jun;17(6):1723-36.
Wu YF, Myasnikova E, Reinitz J. Master equation simulation analysis of immunostained Bicoid morphogen gradient. BMC Syst Biol. 2007 Nov 16;1(1):52 [Epub ahead of print].
Yagil G. Quantitative aspects of protein induction. Curr Top Cell Regul. 1975;9:183-236.
Yagil G, Yagil E. On the relation between effector concentration and the rate of induced enzyme synthesis. Biophys J. 1971 Jan;11(1):11-27.
Yildirim N, Mackey MC. Feedback regulation in the lactose operon: a mathematical modeling study and comparison with experimental data. Biophys J. 2003 May;84(5):2841-51.
Zhu R, Ribeiro AS, Salahub D, Kauffman SA. Studying genetic regulatory networks at the molecular level: delayed reaction stochastic models. J Theor Biol. 2007 Jun 21;246(4):725-45.

Chapter 3 references

de Jong H, Geiselmann J, Hernandez C, Page M. Genetic Network Analyzer: qualitative simulation of genetic regulatory networks. Bioinformatics. 2003 Feb 12;19(3):336-44.
de Jong H, Gouzé JL, Hernandez C, Page M, Sari T, Geiselmann J. Qualitative simulation of genetic regulatory networks using piecewise-linear models. Bull Math Biol. 2004 Mar;66(2):301-40.
Hartwell LH, Hopfield JJ, Leibler S, Murray AW. From molecular to modular cell biology. Nature. 1999 Dec 2;402(6761 Suppl):C47-52.
Inzé D. Green light for the cell cycle. EMBO J. 2005 Feb 23;24(4):657-62.
Inzé D, De Veylder L. Cell cycle regulation in plant development. Annu Rev Genet. 2006;40:77-105.
JavaCC website, https://javacc.dev.java.net
Novak B, Pataki Z, Ciliberto A, Tyson JJ. Mathematical model of the cell division cycle of fission yeast. Chaos. 2001 Mar;11(1):277-286.
Novak B, Tyson JJ. A model for restriction point control of the mammalian cell cycle. J Theor Biol. 2004 Oct 21;230(4):563-79.
SIM-plex website, http://www.psb.ugent.be/cbd/papers/sim-plex
Stals H, Inzé D. When plant cells decide to divide. Trends Plant Sci. 2001 Aug;6(8):359-64.
Thieffry D, Romero D. The modularity of biological regulatory networks. Biosystems. 1999 Apr;50(1):49-59.
Vercruysse S, Kuiper M. Simulating genetic networks made easy: network construction with simple building blocks. Bioinformatics. 2005 Jan 15;21(2):269-71.

Chapter 4 references

Beemster GT, De Veylder L, Vercruysse S, West G, Rombaut D, Van Hummelen P, Galichet A, Gruissem W, Inzé D, Vuylsteke M. Genome-wide analysis of gene expression profiles associated with cell cycle transitions in growing organs of Arabidopsis. Plant Physiol. 2005 Jun;138(2):734-43.
Beemster GT, Vercruysse S, De Veylder L, Kuiper M, Inzé D. The Arabidopsis leaf as a model system for investigating the role of cell cycle regulation in organ growth. J Plant Res. 2006 Jan;119(1):43-50.
Inzé D. Green light for the cell cycle. EMBO J. 2005 Feb 23;24(4):657-62.
MATLAB website, http://www.mathworks.com
MathGrapher website, http://www.mathgrapher.com
Menges M, Murray JA. Synchronous Arabidopsis suspension cultures for analysis of cell-cycle gene activity. Plant J. 2002 Apr;30(2):203-12.
Novak B, Pataki Z, Ciliberto A, Tyson JJ. Mathematical model of the cell division cycle of fission yeast. Chaos. 2001 Mar;11(1):277-286.

Vanneste S, De Rybel B, Beemster GT, Ljung K, De Smet I, Van Isterdael G, Naudts M, Iida R, Gruissem W, Tasaka M, Inzé D, Fukaki H, Beeckman T. Cell cycle progression in the pericycle is not sufficient for SOLITARY ROOT/IAA14-mediated lateral root initiation in Arabidopsis thaliana. Plant Cell. 2005 Nov;17(11):3035-50.

Vercruysse S, Kuiper M. Simulating genetic networks made easy: network construction with simple building blocks. Bioinformatics. 2005 Jan 15;21(2):269-71.

Weijers D, Benkova E, Jäger KE, Schlereth A, Hamann T, Kientz M, Wilmoth JC, Reed JW, Jürgens G. Developmental specificity of auxin response by pairs of ARF and Aux/IAA transcriptional regulators. EMBO J. 2005 May 18;24(10):1874-85.

Chapter 5 references

Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet. 2000 May;25(1):25-9.

Attempto Controlled English (ACE) documentation: http:// attempto.ifi.uzh.ch/site

Baumgartner WA Jr, Cohen KB, Fox LM, Acquaah-Mensah G, Hunter L. Manual curation is not sufficient for annotation of genomic databases. Bioinformatics. 2007 Jul 1;23(13):i41-8.

Blaschke C, Leon EA, Krallinger M, Valencia A. Evaluation of BioCreAtIvE assessment of task 2. BMC Bioinformatics. 2005;6 Suppl 1:S16.

Blaschke C, Yeh A, Camon E, Colosimo M, Apweiler R, Hirschman L, Valencia A. Do you do text? Bioinformatics. 2005 Dec 1;21(23):4199-200.

Chen ES, Hripcsak G, Xu H, Markatou M, Friedman C. Automated acquisition of disease drug knowledge from biomedical and clinical documents: an initial study. J Am Med Inform Assoc. 2008 Jan-Feb;15(1):87-98.

Colosimo ME, Morgan AA, Yeh AS, Colombe JB, Hirschman L. Data preparation and interannotator agreement: BioCreAtIvE task 1B. BMC Bioinformatics. 2005;6 Suppl 1:S12.

Corney DP, Buxton BF, Langdon WB, Jones DT. BioRAT: extracting biological information from full-length papers. Bioinformatics. 2004 Nov 22;20(17):3206-13.

Dickman S. Tough mining: the challenges of searching the scientific literature. PLoS Biol. 2003 Nov;1(2):E48.

Donaldson I, Martin J, de Bruijn B, Wolting C, Lay V, Tuekam B, Zhang S, Baskin B, Bader GD, Michalickova K, Pawson T, Hogue CW. PreBIND and Textomy--mining the biomedical literature for protein-protein interactions using a support vector machine. BMC Bioinformatics. 2003 Mar 27;4:11.

Erhardt RA, Schneider R, Blaschke C. Status of text-mining techniques applied to biomedical text. Drug Discov Today. 2006 Apr;11(7-8):315-25.

Feng C, Yamashita F, Hashida M. Automated extraction of information from the literature on chemical-CYP3A4 interactions. J Chem Inf Model. 2007 Nov-Dec;47(6):2449-55.

Fernández JM, Hoffmann R, Valencia A. iHOP web services. Nucleic Acids Res. 2007 Jul;35(Web Server issue):W21-6.

Gene Ontology Consortium. Gene ontology: tool for the unification of biology. Nat Genet. 2000 May;25(1):25-9.

Gene Ontology Consortium. The Gene Ontology (GO) project in 2006. Nucleic Acids Res. 2006 Jan 1;34(Database issue):D322-6.

Gene Ontology Consortium. The Gene Ontology project in 2008. Nucleic Acids Res. 2008 Jan;36(Database issue):D440-4.

GO, Gene Ontology: http://www.geneontology.org

Hirschman L, Yeh A, Blaschke C, Valencia A. Overview of BioCreAtIvE: critical assessment of information extraction for biology. BMC Bioinformatics. 2005;6 Suppl 1:S1.

Hoffmann R, Valencia A. A gene network for navigating the literature. Nat Genet. 2004 Jul;36(7):664.

Hoffmann R, Valencia A. Implementing the iHOP concept for navigation of biomedical literature. Bioinformatics. 2005 Sep 1;21 Suppl 2:ii252-8.

Hu ZZ, Mani I, Hermoso V, Liu H, Wu CH. iProLINK: an integrated protein resource for literature mining. Comput Biol Chem. 2004 Dec;28(5-6):409-16.

Hu ZZ, Narayanaswamy M, Ravikumar KE, Vijay-Shanker K, Wu CH. Literature mining and database annotation of protein phosphorylation using a rule-based system. Bioinformatics. 2005 Jun 1;21(11):2759-65.

Jensen LJ, Saric J, Bork P. Literature mining for the biologist: from information retrieval to biological discovery. Nat Rev Genet. 2006 Feb;7(2):119-29.

KEGG, Kyoto Encyclopedia of Genes and Genomes: http://www.genome.jp/kegg

Kim JD, Ohta T, Tateisi Y, Tsujii J. GENIA corpus--semantically annotated corpus for bio-textmining. Bioinformatics. 2003;19 Suppl 1:i180-2.

Kim JD, Ohta T, Tsujii J. Corpus annotation for mining biomedical events from literature. BMC Bioinformatics. 2008 Jan 8;9(1):10

Koike A, Kobayashi Y, Takagi T. Kinase pathway database: an integrated protein-kinase and NLP-based protein-interaction resource. Genome Res. 2003 Jun;13(6A):1231-43.

Kinoshita S, Cohen KB, Ogren PV, Hunter L. BioCreAtIvE task1A: entity identification with a stochastic tagger. BMC Bioinformatics. 2005;6 Suppl 1:S4.

Kitano H. Systems biology: a brief overview. Science. 2002 Mar 1;295(5560):1662-4.

Krallinger M, Padron M, Valencia A. A sentence sliding window approach to extract protein annotations from biomedical articles. BMC Bioinformatics. 2005;6 Suppl 1:S19.

Krallinger M, Valencia A. Text-mining and information-retrieval services for molecular biology. Genome Biol. 2005;6(7):224.

Kuhn T, Royer L, Fuchs NE, Schroeder M. Improving Text Mining with Controlled Natural Language: A Case Study for Protein Interactions. DILS 2006, LNBI 4075:66-81.

Leser U, Hakenberg J. What makes a gene name? Named entity recognition in the biomedical literature. Brief Bioinform. 2005 Dec;6(4):357-69.

Medline Citation Counts by Year of Publication: website: http://www.nlm.nih.gov/bsd/medline_cit_counts_yr_pub.html . Retrieved on Feb 4, 2008.

Mewes HW, Dietmann S, Frishman D, Gregory R, Mannhaupt G, Mayer KF, Münsterkötter M, Ruepp A, Spannagl M, Stümpflen V, Rattei T. MIPS: analysis and annotation of genome information in 2007. Nucleic Acids Res. 2008 Jan;36(Database issue):D196-201.

MIPS, Munich Information Center for Protein Sequences: http://mips.gsf.de

Müller HM, Kenny EE, Sternberg PW. Textpresso: an ontology-based information retrieval and extraction system for biological literature. PLoS Biol. 2004 Nov;2(11):e309.

Pyysalo S, Ginter F, Heimonen J, Björne J, Boberg J, Järvinen J, Salakoski T. BioInfer: a corpus for information extraction in the biomedical domain. BMC Bioinformatics. 2007 Feb 9;8:50.

Racunas SA, Shah NH, Albert I, Fedoroff NV. HyBrow: a prototype system for computer-aided hypothesis evaluation. Bioinformatics. 2004 Aug 4;20 Suppl 1:i257-64.

Racunas SA, Shah NH, Fedoroff NV. A case study in pathway knowledgebase verification. BMC Bioinformatics. 2006 Apr 8;7:196.

Reactome, curated database of biological processes in humans: http://reactome.org

Rinaldi F, Schneider G, Kaljurand K, Hess M, Romacker M. An environment for relation mining over richly annotated corpora: the case of GENIA. BMC Bioinformatics. 2006 Nov 24;7 Suppl 3:S3.

Rinaldi F, Schneider G, Kaljurand K, Hess M, Andronis C, Konstandi O, Persidis A. Mining of relations between proteins over biomedical scientific literature using a deep-linguistic approach. Artif Intell Med. 2007 Feb;39(2):127-36.

Rzhetsky A, Iossifov I, Koike T, Krauthammer M, Kra P, Morris M, Yu H, Duboué PA, Weng W, Wilbur WJ, Hatzivassiloglou V, Friedman C. GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. J Biomed Inform. 2004 Feb;37(1):43-53.

TAIR, The Arabidopsis Information Resource: http://www.arabidopsis.org

Tanabe L, Xie N, Thom LH, Matten W, Wilbur WJ. GENETAG: a tagged corpus for gene/protein named entity recognition. BMC Bioinformatics. 2005;6 Suppl 1:S3.

Tsai RT, Chou WC, Su YS, Lin YC, Sung CL, Dai HJ, Yeh IT, Ku W, Sung TY, Hsu WL. BIOSMILE: a semantic role labeling system for biomedical verbs using a maximum-entropy model with automatically generated template features. BMC Bioinformatics. 2007 Sep 1;8:325.

Wheeler DL, Barrett T, Benson DA, Bryant SH, Canese K, Chetvernin V, Church DM, Dicuccio M, Edgar R, Federhen S, Feolo M, Geer LY, Helmberg W, Kapustin Y, Khovayko O, Landsman D, Lipman DJ, Madden TL, Maglott DR, Miller V, Ostell J, Pruitt KD, Schuler GD, Shumway M, Sequeira E, Sherry ST, Sirotkin K, Souvorov A, Starchenko G, Tatusov RL, Tatusova TA, Wagner L, Yaschenko E. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2008 Jan;36(Database issue):D13-21.

Wilbur WJ, Rzhetsky A, Shatkay H. New directions in biomedical text annotation: definitions, guidelines and corpus construction. BMC Bioinformatics. 2006 Jul 25;7:356.

Yeh A, Morgan A, Colosimo M, Hirschman L. BioCreAtIvE task 1A: gene mention finding evaluation. BMC Bioinformatics. 2005;6 Suppl 1:S2.

Yuan X, Hu ZZ, Wu HT, Torii M, Narayanaswamy M, Ravikumar KE, Vijay-Shanker K, Wu CH. An online literature mining tool for protein phosphorylation. Bioinformatics. 2006 Jul 1;22(13):1668-9.

Zhou G, Zhang J, Su J, Shen D, Tan C. Recognizing names in biomedical texts: a machine learning approach. Bioinformatics. 2004 May 1;20(7):1178-90.

Chapter 6 references

Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet. 2000 May;25(1):25-9.

Avraham S, Tung CW, Ilic K, Jaiswal P, Kellogg EA, McCouch S, Pujar A, Reiser L, Rhee SY, Sachs MM, Schaeffer M, Stein L, Stevens P, Vincent L, Zapata F, Ware D. The Plant Ontology Database: a community resource for plant structure and developmental stages controlled vocabulary and annotations. Nucleic Acids Res. 2008 Jan;36(Database issue):D449-54.

Baumgartner WA Jr, Cohen KB, Fox LM, Acquaah-Mensah G, Hunter L. Manual curation is not sufficient for annotation of genomic databases. Bioinformatics. 2007 Jul 1;23(13):i41-8.

Beemster GT, Mironov V, Inzé D. Tuning the cell-cycle engine for improved plant performance. Curr Opin Biotechnol. 2005 Apr;16(2):142-6.

Blaschke C, Leon EA, Krallinger M, Valencia A. Evaluation of BioCreAtIvE assessment of task 2. BMC Bioinformatics. 2005;6 Suppl 1:S16.

Côté RG, Jones P, Apweiler R, Hermjakob H. The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries. BMC Bioinformatics. 2006 Feb 28;7:97.

De Veylder L, Beeckman T, Beemster GT, Krols L, Terras F, Landrieu I, van der Schueren E, Maes S, Naudts M, Inzé D. Functional analysis of cyclin-dependent kinase inhibitors of Arabidopsis. Plant Cell. 2001 Jul;13(7):1653-68.

De Veylder L, Joubès J, Inzé D. Plant cell cycle transitions. Curr Opin Plant Biol. 2003 Dec;6(6):536-43.

Dickman S. Tough mining: the challenges of searching the scientific literature. PLoS Biol. 2003 Nov;1(2):E48.

Harris MA, Clark J, Ireland A, Lomax J, Ashburner M, Foulger R, Eilbeck K, Lewis S, Marshall B, Mungall C, Richter J, Rubin GM, Blake JA, Bult C, Dolan M, Drabkin H, Eppig JT, Hill DP, Ni L, Ringwald M, Balakrishnan R, Cherry JM, Christie KR, Costanzo MC, Dwight SS, Engel S, Fisk DG, Hirschman JE, Hong EL, Nash RS, Sethuraman A, Theesfeld CL, Botstein D, Dolinski K, Feierbach B, Berardini T, Mundodi S, Rhee SY, Apweiler R, Barrell D, Camon E, Dimmer E, Lee V, Chisholm R, Gaudet P, Kibbe W, Kishore R, Schwarz EM, Sternberg P, Gwinn M, Hannick L, Wortman J, Berriman M, Wood V, de la Cruz N, Tonellato P, Jaiswal P, Seigfried T, White R; Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. Nucleic Acids Res. 2004 Jan 1;32(Database issue):D258-61.

Ilic K, Kellogg EA, Jaiswal P, Zapata F, Stevens PF, Vincent LP, Avraham S, Reiser L, Pujar A, Sachs MM, Whitman NT, McCouch SR, Schaeffer ML, Ware DH, Stein LD, Rhee SY. The plant structure

ontology, a unified vocabulary of anatomy and morphology of a flowering plant. Plant Physiol. 2007 Feb;143(2):587-99.
Inzé D. Green light for the cell cycle. EMBO J. 2005 Feb 23;24(4):657-62.
Jaiswal P, Avraham S, Ilic K, Kellogg E, McCouch S, Pujar A, Reiser L, Seung RY, Sachs MM, Schaeffer M, Stein L, Stevens P, Leszek V, Ware D, Zapata F. Plant Ontology (PO): a controlled vocabulary of plant structures and growth stages. Comp. Funct. Genomics 2005; 6:388-397.
Jensen LJ, Saric J, Bork P. Literature mining for the biologist: from information retrieval to biological discovery. Nat Rev Genet. 2006 Feb;7(2):119-29.
Kitano H. A graphical notation for biochemical networks. Biosilico. 2003;1:169-176.
Kitano H, Funahashi A, Matsuoka Y, Oda K. Using process diagrams for the graphical representation of biological networks. Nat Biotechnol. 2005 Aug;23(8):961-6.
Kohn KW. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. Mol Biol Cell. 1999 Aug;10(8):2703-34.
Kohn KW, Aladjem MI, Weinstein JN, Pommier Y. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. Mol Biol Cell. 2006 Jan;17(1):1-13.
Kuhn T, Royer L, Fuchs NE, Schroeder M. Improving Text Mining with Controlled Natural Language: A Case Study for Protein Interactions. DILS 2006, LNBI 4075:66-81.
Menges M, Murray JA. Synchronous Arabidopsis suspension cultures for analysis of cell-cycle gene activity. Plant J. 2002 Apr;30(2):203-12.
Menges M, Hennig L, Gruissem W, Murray JA. Cell cycle-regulated gene expression in Arabidopsis. J Biol Chem. 2002 Nov 1;277(44):41987-2002.
Mironov V V, De Veylder L , Van Montagu M , Inze D. Cyclin-dependent kinases and cell division in plants - the nexus. Plant Cell. 1999 Apr;11(4):509-22.
Motter AE, de Moura AP, Lai YC, Dasgupta P. Topology of the conceptual network of language. Phys Rev E Stat Nonlin Soft Matter Phys. 2002 Jun;65(6 Pt 2):065102.
Nurse PM. Nobel Lecture. Cyclin dependent kinases and cell cycle control. Biosci Rep. 2002 Oct-Dec;22(5-6):487-99.
OLS, Ontology Lookup Service, website: http://www.ebi.ac.uk/ontology-lookup
OWL, Web Ontology Language: overview: http://www.w3.org/TR/owl-features
Pirson I, Fortemaison N, Jacobs C, Dremier S, Dumont JE, Maenhaut C. The visual display of regulatory information and networks. Trends Cell Biol. 2000 Oct;10(10):404-8.
Priedhorsky R, Chen J, Lam SK, Panciera K, Terveen L, Riedl J. Creating, Destroying, and Restoring Value in Wikipedia. GROUP '07: Proceedings of the Association for Computing Machinery conference. 2007;259-268.
Racunas SA, Shah NH, Albert I, Fedoroff NV. HyBrow: a prototype system for computer-aided hypothesis evaluation. Bioinformatics. 2004 Aug 4;20 Suppl 1:i257-64.
SMBL, Systems Biology Mark-up Language: http://www.sbml.org
Stals H, Inzé D. When plant cells decide to divide. Trends Plant Sci. 2001 Aug;6(8):359-64.
Tanabe L, Xie N, Thom LH, Matten W, Wilbur WJ. GENETAG: a tagged corpus for gene/protein named entity recognition. BMC Bioinformatics. 2005;6 Suppl 1:S3.
Vandepoele K, Raes J, De Veylder L, Rouzé P, Rombauts S, Inzé D. Genome-wide analysis of core cell cycle genes in Arabidopsis. Plant Cell. 2002 Apr;14(4):903-16.

Chapter 7 references

Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet. 2000 May;25(1):25-9.
Blaschke C, Leon EA, Krallinger M, Valencia A. Evaluation of BioCreAtIvE assessment of task 2. BMC Bioinformatics. 2005;6 Suppl 1:S16.
Erhardt RA, Schneider R, Blaschke C. Status of text-mining techniques applied to biomedical text. Drug Discov Today. 2006 Apr;11(7-8):315-25.

Ilic K, Kellogg EA, Jaiswal P, Zapata F, Stevens PF, Vincent LP, Avraham S, Reiser L, Pujar A, Sachs MM, Whitman NT, McCouch SR, Schaeffer ML, Ware DH, Stein LD, Rhee SY. The plant structure ontology, a unified vocabulary of anatomy and morphology of a flowering plant. Plant Physiol. 2007 Feb;143(2):587-99.

Jaiswal P, Avraham S, Ilic K, Kellogg E, McCouch S, Pujar A, Reiser L, Seung RY, Sachs MM, Schaeffer M, Stein L, Stevens P, Leszek V, Ware D, Zapata F. Plant Ontology (PO): a controlled vocabulary of plant structures and growth stages. Comp. Funct. Genomics 2005; 6:388-397.

Jensen LJ, Saric J, Bork P. Literature mining for the biologist: from information retrieval to biological discovery. Nat Rev Genet. 2006 Feb;7(2):119-29.

Kim JD, Ohta T, Tateisi Y, Tsujii J. GENIA corpus--semantically annotated corpus for bio-textmining. Bioinformatics. 2003;19 Suppl 1:i180-2.

Kitano H, Funahashi A, Matsuoka Y, Oda K. Using process diagrams for the graphical representation of biological networks. Nat Biotechnol. 2005 Aug;23(8):961-6.

Kohn KW, Aladjem MI, Weinstein JN, Pommier Y. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. Mol Biol Cell. 2006 Jan;17(1):1-13.

Kuhn T, Royer L, Fuchs NE, Schroeder M. Improving Text Mining with Controlled Natural Language: A Case Study for Protein Interactions. DILS 2006, LNBI 4075:66-81.

Krallinger M, Valencia A. Text-mining and information-retrieval services for molecular biology. Genome Biol. 2005;6(7):224.

MineMap, our prototypic community-based ('Biology 2.0') manual text-curation platform: http://www.biology2.net . [user/pass: guest/guest]

OWL, Web Ontology Language: overview: http://www.w3.org/TR/owl-features/

Racunas SA, Shah NH, Albert I, Fedoroff NV. HyBrow: a prototype system for computer-aided hypothesis evaluation. Bioinformatics. 2004 Aug 4;20 Suppl 1:i257-64.

Quote at the beginning of this thesis is by Edmund Hillary; quote at the end by Mark Twain.

Publication List

Peer-reviewed journal publications

(1) K. Himanen, M. Vuylsteke, S. Vanneste, **S. Vercruysse**, et al. (2004). *Transcript profiling of early lateral root initiation.* P.N.A.S. 2004 Apr 6;101(14):5146-51.
(2) P. Hilson, J. Allemeersch, T. Altmann, ..., **S. Vercruysse**, et al. (2004). *Versatile Gene-Specific Sequence Tags for Arabidopsis Functional Genomics: Transcript Profiling and Reverse Genetics Applications.* Genome Research. 2004 Oct;14(10B):2176-89.
(3) **S. Vercruysse** and M. Kuiper (2005). *Simulating genetic networks made easy: network construction with simple building blocks.* Bioinformatics. 2005 Jan 15;21(2):269-71. (Advanced Access Electronic publication: 2004 Aug 19, bioinformatics/bth478)
(4) G.T.S. Beemster, L. De Veylder, **S. Vercruysse**, et al. (2005). *Genome-wide analysis of gene expression profiles associated with cell cycle mode transitions in growing organs of Arabidopsis thaliana.* Plant Physiology. 2005 Jun;138(2):734-43.
(5) A. Verkest, C. L. de O. Manes, **S. Vercruysse**, et al. (2005). *The cyclin dependent kinase inhibitor KRP2 controls the mitosis to endocycle transition during Arabidopsis leaf development through inhibition of mitotic CDKA;1 kinase complexes.* Plant Cell. 2005 Jun;17(6):1723-36.
(6) G.T.S. Beemster, **S. Vercruysse**, et al. (2006). *The Arabidopsis leaf as a model system for investigating the role of cell cycle regulation in organ growth.* J. Plant Res. 2006 Jan;119(1):43-50.
(7) **S. Vercruysse**, M. Kuiper (2008). *MineMap: quality food for the information-hungry biologist.* In preparation, based on chapters 5 and 6.
(8) **S. Vercruysse**, M. Kuiper (2008). *Biology 2.0.* Ready for submission, see chapter 7.

Conference posters

(1) **S. Vercruysse**, M. Kuiper (2004). Simulating genetic networks made easy: Network construction with simple building blocks. At ISMB/ECCB (International conference on Intelligent Systems for Molecular Biology, European Conference on Computational Biology), 2004 Jul/Aug, Glasgow, UK; and at ICSB (International Conference on Systems Biology), 2004 Oct, Heidelberg, Germany.
(2) R. Merks, **S. Vercruysse**, M. Kuiper, G. Beemster, Y. Van de Peer, D. Inzé (2006). From Genes to Organisms via the Cell. Modeling Biological Growth in Plant Science and Biomedicine. Gent-Lille Workshop on 'Computational Biology'. Polytech'Lille. 2006 Jun 20.

Websites

(1) **Lateral root formation**: Website with additional data accompanying the publication (Himanen et al, 2004): http://www.psb.ugent.be/papers/lateralroot
(2) **SIM-plex**: Website with description, tutorial, reference manual, and a download section for the SIM-plex genetic network simulator software: http://www.psb.ugent.be/cbd/papers/sim-plex
(3) **KRP2**: Additional data for the publication (Verkest et al, 2005), with more details about the different models and simulations: http://www.psb.ugent.be/cbd/papers/krp2sim
(4) **MineMap**: Website for cooperative literature curation and exploration, through a Java applet for information entry, display, modification, and graceful visualisation: http://www.biology2.net

20 years from now you will be more disappointed by
the things you didn't do, than by the ones you did do.
So throw off the bowlines. Sail away from the safe harbour...
Explore. Dream. Discover.